ATR Operations Documentation

Editors: G.F. Dell, W.W. Mackay, S. Peggs

Authors: W.W. Mackay, T. Satogata, S. Tanaka, many others

Revision 1.0
October 25, 1995

Contents

Chapter 1

Introduction

1.1 Why read this document?

This document consists almost entirely of World Wide Web pages, that have been printed as postscript files, and compiled into one enormous note.

One advantage of a hard copy of this manual is that it is convenient to carry around, or place in a binder, for easy reference when a computer screen is not readily available. Another advantage is that a tree structure (chapters, sections, subsections) is often easier to search than a web structure. The disadvantage is that the contents may be out of date, especially when things are changing rapidly - for example, when beamlines are being commissioned.

The name of each web page preceeds its entry in this manual. This enables the reader to go back and check for recent updates, by typing the name into a web viewer such as **netscape**. For example, when

/RHIC/ATR/index.html

appears in the text, the real thing can be viewed by entering

http://acnsun10.rhic.bnl.gov/RHIC/ATR/index.html

into the "Location" window in **netscape**.

It goes without saying that the compilers had a relatively easy (if time consuming) task in putting this document together; the real work was done by the authors of the web pages. Therefore, life being what it is, please send all positive comments to the compilers, and address all negtaive criticisms to the authors, whose names (should) appear with their progeny.

Chapter 2

General Information

2.1 Information on the ATR transfer lines

/RHIC/ATR/index.html

This is the "root directory" for many of the ATR pages.

Directory for info on the ATR

- Overview
- Coordinate systems
  - Global coordinate system for surveys
  - Local beamline coordinates
- Definitions and Ack!!-ronyms
- Commissioning & Vertical integration (Limited to AGS and RHIC)
- FEB extraction from AGS
- Installation Report (Database query)
- Safety system (Gates, interlocks, sweeps, ...)
- SiteWideName's (conventions and names)
- Application software

Some rather more esoteric stuff:

- Database descriptions and querries:
  - atr_gddb generic definition database.
  - atr_cal callibration database.

Related Topics

- Brookhaven National Laboratory
- AGS
- RHIC Home page
- RHIC Accelerator Home Page
  - RHIC Accelerator Physics (RAP)
  - Controls
  - RHIC Documentation
  - Installation
  - Instrumentation
  - Tour

Mangled by Waldo MacKay (waldo@bnl.gov).
Last update: 29 Sept., 1995

2.1.1 Overview

/RHIC/ATR/overview.html

Overview of ATR

RHIC is a dual ring collider for ions from protons up to fully stripped gold (100GeV/nucleon).

Over 670 meters of beam line are being built to transfer ions from the AGS to both of the collider rings. These transfer lines are divided into four sections: the U-line which matches beam from the AGS and provides final stripping of gold ions, the W-line which brings the beam along the 6-o'clock-to-12-o'clock symmetry line of RHIC. Starting at the switch magnet just upstream of a beam dump are the two large arcs: the X-line which brings the beam around to the Blue (cw) ring, and the Y-line which brings the beam around to the Yellow (ccw) ring.

2.2 Coordinate systems

2.2.1 Global coordinates

/RHIC/ATR/baseline.html

Global Coordinate System

The RHIC coordinate system is a three dimensional Cartesian system in meters with an origin somewhere above the Atlantic Ocean over forty kilometers to the southwest of RHIC. It is one of several systems used by the Survey and Alignment Group (SAG) at Brookhaven National Lab. The plane of the coordinate system passes through the six interaction regions of RHIC and is not parallel to the plane of the AGS (due to the earth's curvature). Additionally, the RHIC ring is about five feet lower than the AGS. In order to allow the earth's curvature in our optics model, we have effectively kinked (very slightly) the ATR at the two pitching magnets in the w-line in addition to the three ends of the ATR. The actual adjustments were made at the following elements:

- utv1 a slight pitch and roll,
- utv4 a slight pitch and roll,
- wp1 a slight extra pitch in addition to the -12.5mrad pitch,
- wp2 a slight pitch and roll in addition to the +12.5mrad pitch,
- xp1 and yp1 a slight roll and pitch correction.

In order to preserve the alignments of the beamlines with respect to the AGS and RHIC, certain coordinates of the transfer ATR lines have been baselined. For each magnet we define a coordinate called the intersection point (IP) that for a bending dipole is the intersection of the upstream and downstream rays of the design trajectory. For other devices with no bend angle, such as quadrupoles, the IP is defined as the magnetic center of the element. Although a small longitudinal shift of a quadrupole or trim should not change the design trajectory, a change of the IP for a bend or vertical pitch will propagate downstream to other elements. We have decided to baseline the IP's as the simplest way of monitoring that local modifications of the design do not foul up the rest of the beamlines.

A baselined ASCII file containes coordinate information for all the main dipoles, pitching magnets, quadrupoles, and some trim magnets in the U-, W-, X-, and Y-lines. The file consists of one line for each magnet. Each line consists of nine comma-separated fields:

1. survey name of the IP coordinate,
2. N coordinate of IP in RHIC system,
3. E coordinate of IP in RHIC system,
4. W coordinate of IP in RHIC system,
5. theta angle of trajectory cord through magnet relative to the E-axis,
6. phi angle of vertical pitch of cord,
7. psi angle of roll about the cord,

2.2.2 Local coordinates

/RHIC/ATR/beamcoords.html

Local beam line coordinate system

In the ATR transfer lines (U, W, X, and Y-lines) there is a local right-handed coordinate system (*x, y, z*) which moves along with the beam along the design trajectory. The *z*-axis is tangent to the design trajectory pointing downstream. The *y*-axis points up, and the *x*-axis points to the left in the beam's-eye-view which makes the system right-handed.

The cumulative *s*-coordinate for the beam location along the trajectory is measured from the beginning of the U-line (a lattice marker with SiteWideName "ubegin"). The beginning of the U-line (*s*=0) is located along the perpendicular bisector between the AGS dipoles H13 and H14.

For the Blue (clockwise) ring of RHIC, the same convention holds for the local coordinate system with the *s*=0 coordinate at the 6 O'clock crossing. In this case the *x*-axis points radially outward, the *y*-axis upward and the *z*-axis tangentially in the direction of the beam's velocity.

For the Yellow (counterclockwise) ring of RHIC, the convention must be modified, since we want to have the *x*-axis radially outward. The lesser of all evils was determined to be having the *z*-coordinate point in the direction opposite to the beam's motion. The *y* axis is still upward, and the (*x, y, z*)system is still right-handed. The cumulative *s*-coordinate is measured clockwise around the ring, with *s*=0 at the 6 O'clock crossing. With this convention, there is the added advantage that the two rings have *s*-coordinates which propagate in the same direction. Note that for the 4, 8, and 12 O'clock crossings the *s*-coordinates of the two rings differ by almost a meter.

Trim magnet conventions

In the ATR, positive angles in the trim magnets of the ATR should bend the beam in the +*x* direction for horizontal trims, and in the +*y* direction for vertical trims.

In the U-line there are seven trim magnets powered by old monopolar supplies with reversing switches: **psutv1**, **psuth2**, **psuth3**, **psutv4**, **psutv5**, **psuth6**, and **psutv7**. The "A" polarity of these old supplies should bend the beam to the left (+*x*) for horizontal trims and up (+*y*) for vertical trims.

The rest of the trim magnets have bipolar supplies with positive currents bending left (+*x*) and up (+*y*).

Main dipole conventions

In the ATR, for horizontal main dipoles positive angles bend the beam to the right (-*x*). For vertical pitching magnets, positive angles bend the beam downward (-*y*).

All main dipole supplies with the exception of the switching magnet supply **psswm** are monopolar, so that the currents are only positive for **psuarc4**, **psuarc8**, **pswarc20**, **psxarc90**, **psyarc90**, **pswp1**, and **pswp2**. The switching magnet supply is monopolar, but has a reversing switch.

The 100A bias supplies for the lambertson and last dipole magnets of the X and Y-arcs (**psxlamt**, **psylamt**, **psxd31t**, and **psyd31t**) are bipolar and should be wired so that a positive current adds to the positive buss current.

For more information see RHIC/AP/12 W. MacKay and S. Peggs, "Accelerator Physics coordinate conventions".

Mangled by Waldo MacKay (waldo@bnl.gov).
Last update: 3 Oct., 1995

2.3 Lexicon

/RHIC/defs/index.html

Definitions:

ADO
: An ADO is a program which runs on a front end computer which understands and talks to a piece or pieces of hardware. Some useful documentation includes:
  - ADO Classes
  - ADO Events

AdoIf
: The AdoIf is a library of functions to interface program on the console with ADO's running on front end computers.

ATR
: The AGS to RHIC beam transfer lines including the u, w, x and y beamlines.

BCM
: The beam current monitors are circular ferrites place around a ceramic break in the beam pipe. The monitor acts as a transformer with the beam acting as a primary winding and a secondary winding wound around the ferrite ring.

BLM
: The beam loss monitors are gas filled ionization chambers which are placed along the beam line near the beam pipe.

BPM
: Beam position monitors are stripline detectors for measuring the transverse location of the beam within the beam pipe. More info on BPM"s

CLC (Console level computer)
: A console level computer is a workstation (Typically a SUN) which is set up to run application codes with the environment of the main control room (MCR).

FEC (front end computer)
: A front end computer is a controller of a VXI/VME crate running VXWorks and is connected via the network to the console computers. The front end computer talks to various pieces of hardware.

MCR (Main Control Room)
: The Main Control Room is the Main AGS Control Room in Building 911. The ATR commissioning tests will be run from the MCR.

PET
: PET is a parameter editing tool which can be run on a console. It is an engineering level parameter page program with some plotting capabilities. It has been designed to talk to ADO's. The pages can be configured by ASCII files. Some useful documentation includes:
  - PET file format

PLC
: A type of computer used to perform low level controlling and monitoring of

hardware, such as power supplies, vacuum controllers, and radiation safety systems. Danged if I know what the acronym stands for! Maybe something like "Peculiar little computer".

RTDL
: Real Time Data Link.

RSC
: Radiation Safety Committee for the AGS and RHIC. It is currently chaired by Ken Reece.

VPM
: VPM (Video Profile Monitor) refers to the combination of flag, camera and vidio frame grabber system for measuring profiles of the beam at twelve different locations in the ATR

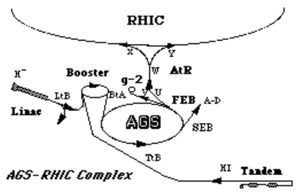
VXWorks
: VXWorks is a real-time version of the Unix operating system which runs on front end computers.

Mangled by Waldo MacKay (waldo@bnl.gov)
Last update: 7 Sept., 1995

2.4 FEB extraction from AGS

//www.ags.bnl.gov/ tanaka/agsfeb.html

The AGS *New* FEB Extraction



Welcome to the AGS FEB Home Page atagsfeb.html the BNL AGS Accelerator Complex. The *new* FEB extraction system performs mutiple sigle bunch extraction (MSBE) of either a heavy ion beam for the Relativistic Heavy Ion Collider(RHIC) through the AtR Transfer Line or a high intensity proton beam for the muon g-2 Experiment(E821) at a rate of 30 Hz up to 12 times per AGS cycle.

General Information

- Introduction to FEB
- AGS Complex Chronicle(.ps)
- FEB Snapshots
- FEB Reports and Documentation

Activities in FY96

●FEB Information (meetings, news etc.)
●FEB(AtR) Commissioning & Run with Au77+
●FEB(V) Commissioning & Run with protons

---

sanki@bnl.gov
Updated:10.Oct.95

2.5 ATR commissioning

2.5.1 Beam requirements

/RHIC/ATR/vip/beamrequire.html

Beam Requirements for 1995 Commissioning

Ion species from AGS:	Gold (A=197, Z=79) Charge: +77 (2 electrons left)
Momentum:	>28.00*(Z/A) [GeV/c]
Intensity:	1x10^10 charges/bunch
Normalized emittance:	10pi [mm.mr]
Bunch length (95%):	~20ns (~7 ns with bunch rotation)
Bunch area (95%):	0.2 eV.sec/u (with bunch rotation)
dp/p:	~0.001
Timing signals:	30 +/- 0.1 microseconds before kicker
Operation mode:	Context switching with HIP/SEB roughly 1 every 10 to 20 cycles Generally only one bunch per FEB cycle
h(rf):	either 8 or 12
Running period:	Mid October through mid December

Last update: 19 Sept, 1995
Mangled by Waldo MacKay (waldo@bnl.gov).

2.5.2 Commissioning strategy

//RHIC/ATR/vip/comstrat.html

The AGS Operations Procedure Manual, AGS-TPL 95.10, is included in Appendix A

Commissioning Strategy for Fall 1995

The following rough outline quite possibly covers more than may be achieved in the two+ month period allotted for commissioning the line. In fact the absolute minimum which should be completed next fall is to transport beam reliably to the beam dump, perform the fault studies and satify the Radiation Safety Committee. We would hope that more could be done, although methods can be improved and retried during future running periods.

1. Things to do before beam tests
   1. check cooling water on magnets
   2. ramp magnets
   3. check polarities of magnets
   4. pump down line and check vacuum
   5. check interlocks
   6. check other hardware
      1. BPM's: cables and electronics.
      2. BLM's.
      3. Flags: read back pictures with calibration lights.
      4. Collimators: check motor control and location read-backs.
      5. Current transformers and electronics.
      6. Timing system: check signals.
   7. Test software and control system.
   8. Test safety system.
   9. Get documentation and training procedures in order.
   10. Pass all the review hurdles.
   11. Verify that all training has been preformed.
   12. Check new AGS features (with machines studies this spring, if and where possible).
       1. Extraction bumps around H10 and G10.
          1. Check the predicted knobs with proton beam.
          2. Measure tunes versus amplitude.
          3. What are the effects of sextupoles and bump amplitude on the beam?
       2. Demonstrate extracttion with fast kicker FKG10 for SBE (single bunch extraction) tests to B-line.
       3. Upgrade at least 5 PUE's (BPM's to the rest of the world) for sensitivity to gold beam.

        4. Install the H10 septum magnet this summer.
2. With beam
    1. Thread beam down the U- and W-lines.
        1. Steer the beam onto the flags.
        2. Measure the location with the BPM's.
        3. Verify magnet and BPM polarities with beam.
        4. After reaching a flag with a reasonable trajectory, remove the flag and go on to the next one.
    2. Measure the pulse stability from the AGS.
        1. Beam current.
        2. Position.
        3. Profile on flags.
    3. Do fault studies.
        1. Check for radiation leaks when the beam hits certain key elements. Of particular interest are:
            1. Access doors, particularly in the split region.
            2. Penetrations for ventilation shafts and cables.
            3. Thin shielding areas.
            4. The top of the berm where Thompson road crosses the beam line.
    4. Measure the transverse matrix elements (*C*, *S*, *C'*, *S'*) for both *x* and *y*.
        1. Measure the beam location at all BPM's.
        2. Change **utv1** by a small amount and remeasure the trajectory.
        3. Reset **utv1** to previous value and remeasure the trajectory.
        4. Change **uth2** by a small amount and remeasure the trajectory.
        5. Calculate the expected deviations and compare with data.
    5. Measure the dispersion elements of the beam line (*D*, *D'*).
        1. Measure the trajectory.
        2. Simulate a 0.1% momentum change by ramping all magnets up by 0.1%.
        3. Remeasure the trajectory.
        4. Calculate the values of *D* and *D'* at the BPM locations.
        5. Compare with the expected values.
    6. Measure the beam shape (hyperellipsoid)
        1. Measure the profile at flags **uf3**, **uf4**, and **uf5**
        2. Measure the profile at flags **wf1**, **wf2**, and **wf3**
        3. Attempt to measure momentum spread with collimator **uc1**.
        4. Calculate emittances, betas, and alphas (horiz and vert) at the flag locations.
    7. Measure dispersion of the beam.
        1. Change the momentum of the AGS extracted beam.
        2. Remeasure the trajectory.
        3. Calculate the values of dispersion functions (eta, eta') at the BPM

        locations.
    4. Compare with the expected values.
  8. Tune the U-line quads to best match the desired values going into the W-line.
     - Note that the dispersion should be zero at the entrance to the W-line (20 degree arc).
  9. Tune the W-line quads to best match the desired values just upstream of **swm** (switch magnet).
     - Note that the dispersion just upstream of the switch magnet should also be zero.
  10. Scan aperture

3. Things to do beyond the 1995 tests.
   1. Transport beam down the Y-line for the 1996 sextant test.
   2. Strip last electrons from gold ions.
   3. Timing signals for injection kickers.
   4. RHIC abort signal connections.
   5. Transport beam down the X-line and inject into the Blue ring.
   6. ...

2.6 Vertical Integration Projects (VIP)

2.6.1 VIP index

/RHIC/ATR/vip/index.html

This is the "root" page for the following set of web pages:

System and stuff for ATR Commissioning

(*Warning:* The following information is somewhat volatile, and should be used with caution.)

- Overview of Commissioning.
  - Definitions.
  - 1995 Beam Requirements
  - Schedule
  - Minutes and memos.
- SiteWideName's
- FEB extraction from AGS
- Magnet systems (power suppies, WFG's, ...)
- Injection kickers.
- Timing.
- Monitors
  - Beam threading (BPM's, ...)
  - Beam Loss Monitors.
  - Current transformers
  - Beam profile measurement system (flags, ...)

Nothing yet in the following:

- Collimators

Mangled by Waldo MacKay (waldo@bnl.gov)
Last update: 7 Sept, 1995

2.6.2 RHIC injection kickers

/RHIC/ATR/vip/kickers/index.html

RHIC Injection kickers

The overall performance requirements are:

Performance Specification

Deflection angle	1.86 mradian (total for 4 magnets)
Beam rigidity	97.5 Tm
Rise time (1-99%)	95 ns
Flat top	20 ns
Flat top tolerance	+/-1%
Fall time	<800 ns
Minimum repetition	33 ms
Lifetime	>1,000,000 pulses

The individual magnet characteristics are:

Strength at 2000 A	0.0465 Tm
Propagation time	45 ns
Impedance	25 Ohms
High frequency cutoff	~25 MHz
Magnet aperture	48.4 mm wide by 52.1 mm high
Magnet length	1.12 m
H field deflection	~94%
E field deflection	~6%
Ceramic beam tube ID	41.3 mm
thickness	32 mm wall
Core material	
Ferrite	Ceramic magnetic CMD 5005
	15 sections: 50 mm long x 13.9 mm thic
Dielectric	Trans-Tech MCT - 100
	14 sections: 25 mm long x 13.9 mm thic
Bus bar/return frame	6061-T6 aluminum
Epoxy potting material	Conap Inc. RN1000

Mangled by Waldo MacKay (waldo@bnl.gov)

Last update: 7 Apr., 1995

2.6.3 ATR magnet tables in atr_gddb

/RHIC/ATR/vip/magnet/magnet_tables.html

There are several tables in the atr_gddb database which deal with magnets. All units are assumed to be in SI units (Amperes, meters, Tesla, ...).

First there is the *magnet_slot* table which contains information relative to the lattice, such as the SiteWideName, *GenericName*, and *SerialName* of the magnet or other beamline element within that slot. It does not contain information on vacuum components or BLM's. The *SiteWideName* is the primary key for this table. The colums are as follows:

```
SiteWideName:        SiteWideName of the element,
GenericName:         GenericName of the element, such as a model na
SerialName:          Actual serial id of what element is installed,
Orientation:         This specifies the direction of installation o
                     element: +1 for forward, -1 for reversed.  The
                     elements currently reversed are some dipoles.
sag_correction:      A transverse offset of magnet's center from de
                     trajectory to account for bending angles.
IP_fixed:            A flag for indicating that the IP has been def
```

Certain information which is specific to a particular element SerialName can be found in the *magnet_data* table. The primary key for this table is the *SerialName* which can be linked with the SerialName column of the *magnet_slot* table. The columns are as follows:

```
SerialName:          actual serial name of a particular device,
ftname:              link into "fid_offsets" table for generating
                     fiducials relative to the IP coordinates,
FieldName:           link into the "magnet_field" table for transfe
                     function information of a particular kind of m
long_corr:           a longitudinal correction which allows for a s
                     misalignment of the fiducial template used in
                     construction of some dipole magnets,
top_halfcore:        serial name of the top halfcore of a dipole ma
bottom_halfcore:     serial name of the bottom halfcore of a dipole
```

The two tables *magnet_slot* and *magnet_data* make up the *magnet_info* group of tables. An SDS file containing the data of these two tables may be generated by the command:

```
dbg2sds atr_gddb magnet_info
```

The view *magnet_survey* is a view of selected columns from the tables *magnet_slot* and *magnet_data*. This view is used by the program **atrfid** to generate survey fiducials for the beamline elements. Its columns are as follows:

```
SiteWideName:        SiteWideName of the element,
```

ftname:	link into "fid_offsets" table,
Orientation:	flag for reversed dipole stands,
long_corr:	a longitudinal correction which allows for a s misalignment of the fiducial template used in construction of some dipole magnets,
IP_fixed:	A flag for indicating that the IP has been def

Together with the *fid_offsets* table, the *magnet_survey* view form the group *fidgen* of tables/views which is used by the **atrfid** program to generate fiducials for the surveyors.
The columns of the *fid_offsets* table are:

ftname:	name of group of fiducial offsets which links back to SerialName in the "magnet_data" table,
fidname:	name of fiducial to be combined with a SurveyName to generate a specific fiducial survey name as used by th Survey and Alignment Group,
dx:	horizontal transverse offset of the fiducial,
dy:	vertical offset,
dz:	longitudinal offset.

A complete list of fiducials which have been generated may be found in file $HOLY_LATTICE/ATR_common/fid_mike.list.
These fiducials may be by the **plot_atr** program with the command:

```
   plot_atr -geometry 636x873+450+1 $HOLY_LATTICE/ATR_common/fid_mike.
```

With the mouse-cursor inside the graphed window, type "h" and a help message will give a list of possible commands for using this rather eccentric program.

Transfer functions for the various magnets are given by the *magnet_field* table whose columns are:

FieldName:	key to select a given type of magnet data
sequence:	sequence number for data
I:	current in amperes
Transfunc:	Bl/I for dipoles, Gl/I for quads
RefRadius:	reference radius for multipole components (in
UpDown:	direction of ramp =0 for simulation, =+1 for up, =-1 for down
b0:	Normal dipole multipole (=1 for dipoles, =0 fo
b1:	Normal quadrupole multipole (=1 for quads)
b2:	Normal sextupole component
b3:	Normal octopole component
b4:	Normal decupole component
b5:	Normal 12-pole component
a0:	Skew dipole component
a1:	Skew quadrupole multipole (=1 for quads)
a2:	Skew sextupole component
a3:	Skew octopole component

a4:	Skew decupole component
a5:	Skew 12-pole component

The definition of transfer function is the integral of B ds divided by the current. In a first approximation, the transfer function is a constant, but it more accurately drops off at high field due to saturation effects. All units are assumed to be in SI units (Amperes, meters, Tesla, ...).

A group of views *magfield* has been defined to quickly access the relevant information. This group consists of the views: *magquick* with the columns (from the tables *magnet_slot*, *magnet_field*, and *NameLookup*):

SiteWideName:	SiteWideName of the magnet from NameLookup
lattice_index:	Pointer into the Holy_Lattice flat SDS file
FieldName:	key into the "magquick" view for the transfer function of the magnet.
Scoord:	The s-coordinate from the beginning of the U-l

magfield with the columns from *magnet_field*:

FieldName:	key to select a given type of magnet data
sequence:	sequence number for data
UpDown:	direction of ramp =0 for simulation, =+1 for up, =-1 for down
RefRadius:	reference radius for multipole components (in
I:	current in amperes
Transfunc:	Bl/I for dipoles, Gl/I for quads
b0:	Normal dipole multipole (=1 for dipoles, =0 fo
b1:	Normal quadrupole multipole (=1 for quads)

and the view *ufoil* with the single entry

Scoord:	the s-coordinate of the stripping foil in the

This *magfield* group may be dumped into an SDS file by the command:

```
   dbg2sds atr_gddb magfield
```

There is a function **magfield_in_sds()** which may be used to read in the data from this SDS file as shown in the example code in the file $HORST/ATR/magnets/magfield_test.c. The SDS object structures are defined by the include file $HORST/include/gddb/magfield.h. In order to link the correct libraries with **gcc** you need to include something like

```
        -L $(HORST)/lib/$(ARCH) -lgddb \
        -L $(ISTKPLACE)/lib/$(ARCH) -lsds \
        -lm
```

in the command line.

Mangled by Waldo MacKay (waldo@bnl.gov)
Last update: 16 May, 1995

2.6.4 Current transformers

/RHIC/ATR/vip/transformer/index.html

ATR Beam Intensity Measurement

The charge in each bunch transferred from the AGS to RHIC will be measured at five strategic points along the transfer line. These are: (1) just after extraction from the AGS; (2) just after stripping and collimation; (3) immediately before entry into one of the two injection arcs; (4) at the end of each arc just before injection into RHIC.

The detectors will be current transformers, specifically the Integrating Current Transformer (ICT) type developed by Unser[1] for LEP. This design incorporates two toroidal transformer cores and is particularly suited to measuring the charge in short beam bunches. By passively stretching the initial pulse, the signal is converted downward in frequency prior to applying it to the transformer. This overcomes the core losses at high frequencies, while preserving the relative charge and, in addition, allows the electronics to be located further from the detector. While the temporal beam signal is lost in this technique, precision measurements are made possible over a wide range of beam intensities.

Each ICT will be mounted over a ceramic break in the beam pipe and covered by a copper enclosure to provide a path for the beam induced wall currents. Triaxial cable will be used to bring the signal to the electronics, which is as far as 150 meters away in one case. The use of this kind of cable will allow the transformer outputs to remain isolated electrically from the beam pipe and building grounds and yet provides for adequate shielding of the signal conductors in the presence of noise generated by nearby high current beam kickers and other devices.

The signal processing electronics needed for the ICT signal will be resident in the Eurocrates which also contain the BLM electronics. Processing consists of an accurately timed and compensated integration of the stretched charge signal and subsequent digitization by an analog to digital converter.

[1] K. B. Unser, "Measuring Bunch Intensity, Beam Loss and Bunch Lifetime in LEP", Proc. IEEE Particle Accelerator Conference EPAC 90, Nice, France, June 1991.

2.6.5 Beam loss monitors

/RHIC/ATR/vip/blm/index.html

Loss Monitors

excerpt from the RHIC Design Manual (April 1994)

Beam Loss Monitor System

The main functions of the Beam Loss Monitor (BLM) system for RHIC are:

1. To provide an abort signal to avoid quenching the superconducting magnets.
2. To provide a history (postmortem) of the losses preceding an abort to help identify the source of the problem.
3. To provide spatial and temporal loss data to assist in tuning the beam to reduce the losses. The RHIC BLM System will not be used for personnel protection.

The quench thresholds are taken as 2 mJ/g for "fast" losses and 8 mW/g for "slow" losses, where "fast" and "slow" are relative to the time-constant (100 msec) with which the cryogenic system can remove heat from the coils.

Data from the BLM system will be stored in a circular buffer which will stop on an abort to help diagnose the fault which led to the beam dump. The loss data in local memory will cover a period of about 10 sec, comparable to the BPM data.

The detectors will be placed at locations where they will be most sensitive to beam loss which might quench the magnets. The average spacing will be about 15 m. Relocatable units will be placed near injection or extraction equipment, or at temporary problem areas, where control of losses is especially critical. The distribution of detectors is shown in table 3. The electronics will be located in 24 alcoves and houses around the RHIC tunnel in the same racks as the BPM electronics. By using a modular design significant system expansion can be provided by adding more VXI/VME cards or crates.

Table 3: Distribution of Detectors

Location	Number Per Location	Total
Standard Triplet	6	72
Standard FODO	7	84
Standard Arc	23	138
rf Region	8	8

Collimators/Scrapers	2	8
Snake/Spin Rotators	2	24
Beam Dumps	8	16
Injection Region	6	12
Relocatable Units	20	20
Total		382

Determining which beam was the source of the radiation could be useful but would not prevent both beams from being dumped. Since individual detectors will not have directional characteristics, it may be possible to use signals from adjacent detectors to determine directionality. It has been estimated that four decades of dynamic range would be required. This will be taken as the design goal but will not be a design requirement.

An ion chamber of the type used successfully on the Tevatron will be used as the detector. After conversion of the signal from a current to a voltage, the output will be processed in several ways. It will be directly available as an analog signal via the multiplexer. The time response for electron collection will be comparable to a single turn. The output of the front-end amplifier will also go to an integrator which will accumulate data over several turns before being digitized and placed into the circular buffer. The signal to the integrator will be filtered to correspond to the digitizing rate. This will be necessary if the four decade range is to be achieved.

The signal will also be fed through a circuit block which simulates the magnet quench time response and then to a comparator which will generate an abort signal on crossing a reference level. Any of the 382 detectors will be able to trigger a beam dump. However, since some detectors will be located near higher loss locations, levels will be individually adjustable for each detector. Up to 16 event drivel levels will be available which will allow for energy increase and operational flexibility. There will be separate trip levels for fast and slow losses. The slow loss abort may be generated from data in the local memory. Each unit will be able to be masked to allow higher loss during studies or to disable bad channels. An indication of which detector and loss mode triggered the abort will be available in a status word. A means of checking the abort system calibration will be provided.

NAME
     moveblm.sh -- script for updating positions of movable BLM's
     in atr_gddb

SYNOPSIS
     moveblm.sh

This script updates the atr_gddb..otherelement database table for SiteWideName's which are listed in the atr_gddb..movableBLM table. It prompts the user for the relevant information. It should be run from the "mcr" account which has privileges to update this table.

A useful command line to use with pagetree would be something like:

```
  xterm -geometry 165x32+1+1  -e moveblm.sh&
```

KNOWN BUGS

SEE ALSO
     pagetree

2.7 Timing

/RHIC/ATR/vip/timing/index.html

THIS IS A COMPLETE REVISION

OF

RHIC REPORT AD/RHIC/RD-16 August 1993

last revision July 24, 1995

1.0 RHIC EVENT TIMELINE SYSTEM - RHIC EVENT TIMELINE SYSTEM

1. 1 INTRODUCTION

A modern accelerator must synchronize the operations of equipment over a wide area. To facilitate this synchronization the RHIC event system will provide a highly reliable, serial timing link to all equipment locations. Timing events and clocks from this link will be used to initiate hardware operations including changes in settings, state changes, and data acquisitions. Events may also be required by software running on systems not directly coupled to accelerator hardware. A standard clock frequency of 10 MHz, as presently used in the AGS and Booster, will provide adequate resolution for timing events in the RHIC acceleration and collision processes.

Two mechanisms are available for generating events on the RHIC timeline, direct hardware inputs and software initiated commands. Optically isolated TTL-level inputs are provided for each of the 256 possible event trigger inputs. Event sequences to initiate waveforms, fire kickers, and acquire data during the acceleration cycle, tune measurements, etc. will be implemented by cascading programmable delays. Clocks that are of a general interest, such as the 720 Hz clock generated by the main magnet power supply system will also be available on the RHIC timeline. Externally generated events may also come from other systems sensing unusual conditions with the beam. In the case of a beam abort, the abort event can be used to freeze circular buffers in digitizers for postmortem analysis.

RHIC event codes can be permanently assigned without regard to their timeline transmission priority level. The V100 event encoder module contains an event trigger to event code translation table. This table allows an event code trigger priority level to be adjusted relative to other event codes without changing, the event code.

An example of a software generated event would be one to activate new settings after they have been loaded and verified. Software generated events also provide a convenient way to commission new systems.

The probability exists that several event requests could occur simultaneously, or overlapped in time. Since only one event can be processed at a time, priority resolution is an integral part of the central encoding facility. Event contention is handled in hardware with highest priority given to input 0 and lowest given to input 255. It should be pointed out, however, that lower priority events being processed will not be interrupted by the arrival of a higher priority event request.

The RHIC central event encoder will be located in the 4 o'clock equipment house (currently located in the AGS 911b equipment room). The V100 event encoder, its V101 input modules, and supporting front end computer interface will occupy a single VME chassis. Each V101 input module can support 16 event trigger inputs. The event system interconnections are point-to-point, differential TTL. The V100 event encoder module is isolated from its receiver by transformer coupling at the receiver. The V100 event encoder initially drives a D09-E2440 fanout assembly which provides 16 buffered, TTL differential outputs. Some outputs will be used locally within the 4 o'clock house, and others will drive fiber-optic transmitters for transmission to remote

RHIC equipment locations.

At each remote RHIC equipment location, the optical transmission is converted to single-ended TTL, and buffered as differential TTL. Again the D09-E2440 fanout assembly is used to produce multiple outputs. General purpose V102 delay modules may be located in these remote locations, as well as other modules having direct timeline inputs (e.g. the waveform generator).

The V102 delay modules accept the RHIC event timeline transmission, decode selected event codes, and generate TTL compatible event pulses following individually programmable delay intervals. The module outputs have 50 ohm drive capability. It is recommended that equipment using these event pulses provide optical isolation to eliminate ground loops.

The event codes are transmitted using a serial modified Manchester code (bi-phase-mark). The transmission rate is 10 Bits/sec, and 1.2 usec are required to transmit an event code. The event timeline contains a continuous bi-phase-mark "ones' transmission during idle periods. A single event code transmission is shown in

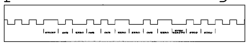
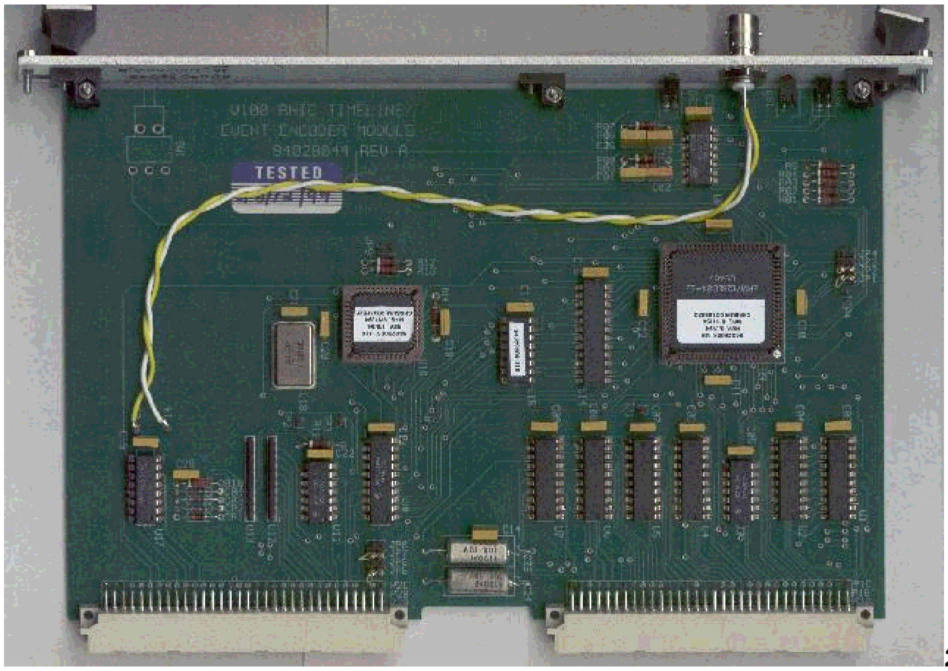


figure.

There is a 1.3 usec event code transmission delay built into the event encoder system. This delay allows very high priority events will be transmitted with minimum time jitter (-O, + 100 nsec). If a higher priority event trigger is received while a lower priority event trigger is being delayed, the delay is reset, and the high priority event code will be transmitted before the low priority event code (with its own 1.3 usec delay).

2.0 V100 EVENT ENCODER MODULE

Reference: 94028046 Module Assembly - V100 RHIC Timeline Encoder Module



2..1 Introduction

The V100 event encoder module is used to transmit RHIC event codes on the RHIC timeline. The encoder module is connected to a 16-position event input module bus. V101 event input modules determine the relative priority of its 16 event trigger inputs. If triggered, the V101 event input module delays 1.3 usec, and attempts to place its highest priority event trigger on the input module bus. However, the V101 event input modules are part of a serial, daisy chain, input module priority scheme. Thus, if other V101 event input modules have been triggered, the V101 event input module must have position priority (closest to the encoder module) in order to place its event trigger code on the bus.

The event trigger with the least numeric value has the highest priority. The V100 event encoder module converts event trigger inputs into RHIC event codes in a translation table. The event code is input to a serial non-return-to-zero to bi-phase-mark converter for transmission on the RHIC timeline.

When the V100 event encoder module receives an event trigger available from an V101 event input module the following occur:

- The event trigger code is loaded into a temporary buffer.
    - The event trigger code translated into an event code.
        - The translation table is memory mapped on the VMEbus. The translation table is available to the front end computer, except during event trigger to event code translation. The translation delay is transparent to the front end computer.
- The V100 event encoder module sends a BUSY signal to the highest priority V101 event input module, disabling the priority chain

while the transmission is in process.

- The translated event code is input to a serial non-return-to-zero to bi-phase-mark converter. As the code is serialized, a 'zero" start bit is added, and code parity (even) is generated. The output of the bi-phase-mark converter is transmitted over the RHIC timeline.
- The BUSY signal is terminated at the end of the event code parity bit transmission. This allows the V101 event input module priority system time to determine the next event trigger (if any) to be transmitted by the encoder module, and maintain minimum headway.
- The minimum inter-event code transmission headway is two bit times (two bi-phase-mark "ones").

2.2 V100 Encoder Module Front Panel

- VME SEL:
  - Green LED, indicates that the encoder module has been addressed as a VMEbus slave, read or write. Pulse stretched for visibility.
- OFFLINE:
  - Red LED, indicating that the front end computer has not enabled the module. The timeline carrier is output, but no event code can be transmitted.
- OUTPUT CONNECTOR:
  - Connector label EVENTLNK.
  - Amphenol 3-2225 twin-axial connector
  - EVENT
  - Green LED, indicating a timeline event code transmission. Pulse stretched for visibility.

2.3 V100 Event Encoder VMEbus Interface Details

2.3.1 V100 Event Encoder Module BASE_ADDRESS

The V100 event encoder module BASE_ADDRESS sets the base address for the event encoder system (encoder module, and 16 V101 event input modules). The V100 event encoder module base address is set in a 4-bit jumper patch. The simple VMEbus A16 decoding requires that the BASE_ADDRESS be a multiple of 0x1000, and that 0x1000 bytes are reserved for the event encoder and its input modules.

```
______________MSB___________ADDRESS-16____________LSB
_____________|15|14|13|12|11----------------------00|
BASE_ADDRESS |_X|_X|_X|_X|_0------------------------0|
```

The first 0x800-bytes are reserved for the encoder module. Only 0x200-bytes are used, but simple address decoding prevent the use of the remaining 0x600-bytes for other purposes. If addressed between BASE_ADDRESS + 0x200-0x7FF, the event encoder will respond, however the results are undefined. Also, if addressed between BASE_ADDRESS + 0x42-0xFF (CSR and SRAM) the results are undefined.

Setting the encoder module base address also sets the input module base address. Each input module requires 0x80 bytes. As input modules are added to the system, each module will add an 0x80 bytes to the event encoder system response range (16 input modules maximum).

2.3.2 V100 Event Encoder Module VMEbus VMEid Message

Located at the BASE_ADDRESS is the 64-byte, read only, VME module VMEid message. Odd bytes contain the message, even bytes contain an ASCII ".".

2.3.3 V100 Event Encoder Module Command and Status Register

The command and status register is located at BASE_ADDRESS + 0x41. The register contains a single read-write bit; GO. GO is set to allow the encoder module to accept event triggers from the input module bus. If GO is reset, the front panel OFFLINE LED is illuminated.

If GO is reset, the V100 encoder module appears BUSY to the V101 input modules. If the input modules are enabled, event triggers can be processed, but not transferred to the encoder. As a result, triggered input modules will soon interrupt the front end processor.

```
______________MSB--------------------------LSB
CMD_STAT_REG_|_0_|_0_|_0_|_0_|_0_|_0_|_0_|_GO|
```

2.3.4 V100 Event Encoder Module Event Code Translation Table

Event code translation table is located at BASE_ADDRESS + 0x100. The table converts input module event triggers to event codes for transmission on the timeline. The table address is the event trigger input module channel. The highest priority event trigger is table address 0x00. The contents of the table is the timeline event code. The V100 event encoder module Event Code Translation Table

```
______________MSB-------EVENT CODE--------LSB
EVENT_IN_T00_|_e_|_e_|_e_|_e_|_e_|_e_|_e_|_e_| BYTE_0 highest priority
```

```
EVENT_IN_T01_|_e_|_e_|_e_|_e_|_e_|_e_|_e_|_e_| BYTE_1
EVENT_IN_T02_|_e_|_e_|_e_|_e_|_e_|_e_|_e_|_e_| BYTE_2
EVENT_IN_T03_|_e_|_e_|_e_|_e_|_e_|_e_|_e_|_e_| BYTE_3
"
"
EVENT_IN_TFF_|_e_|_e_|_e_|_e_|_e_|_e_|_e_|_e_| BYTE_FF lowest priority
```

where: EVENT_IN_Txy; x = V101 input module; y = V101 input module channel; and e = event code

2.4 V100 Event Encoder Jumper Setup Details

The V100 event encoder module sets the BASE_ADDRESS of the V100 event encoder module, and its supporting V101 input modules. The V100 encoder module also sets the most significant nibble of the V101 module STATUS_ID (interrupt vector). The least significant nibble is the V101 input module input bus position.

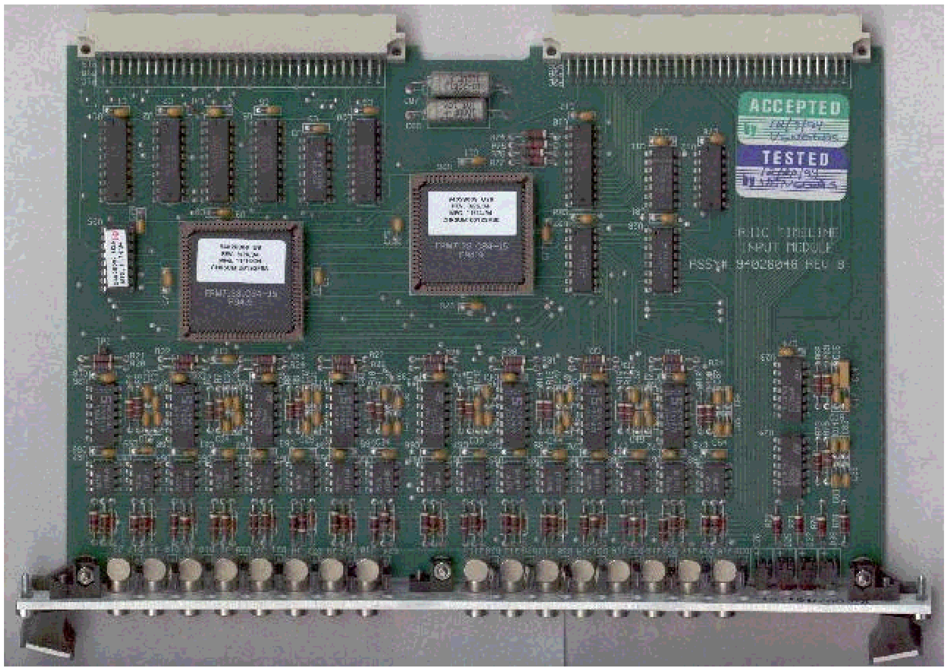
V100 JUMPERS		
LOCATION	**JUMPER**	**FUNCTION**

JP[1..4]	ON = ZERO
	OFF = ONE
		VMEbus A16 ADDRESS A[15..12]

JP[5..8]	ON = ZERO
	OFF = ONE
		INTERRUPT VECTOR STAT_ID[7..4]

3.0 V101 EVENT INPUT MODULES

Reference: 94028050 Module Assembly - V101 RHIC Timeline Input Module



3.1 Introduction

Each V101 event input module accepts up to 16 event trigger inputs, and determines their relative priority. V101 event input module channel 0 has the highest priority and channel 15 the lowest priority. Sixteen input modules are required to support 256 events.

Event triggers can originate from two sources:

- V101 event input module trigger input:
  - Front panel LEMO NIM/CAMAC connectors
  - VMEbus P2 connector user pins
- VMEbus write to the V101 event input module trigger register

It is possible for a low priority trigger to be delayed by higher priority triggers. In this case there may be several transmission increments (1.2 usec/increment) from the trigger until the event is transmitted on the RHIC timeline.

As a result of event trigger priority processing, it is possible to have an extended delay before a low priority event trigger has been processed. Therefore, it is possible for a low priority channel to be retriggered before a previous trigger is processed. In this case the V101 event input module will detect the error, and set a flag in the V101 error register. The error flag indicates that one or more event triggers have been lost. If the channel interrupt is enabled, the error will generate a VMEbus interrupt. The interrupt status/ID vector is derived from a 4-bit jumper patch on the V100 Encoder Module, and the V101 Input module position on the input module bus. The V101 VMEbus interrupt request level is programmable. During interrupt service, the event front end computer can determine which event trigger channel caused the interrupt.

3.2 V101 Input Module Front Panel

- EXTERNAL TRIGGER INPUTS:
  - 16-LEMO NIM-CAMAC connectors
    - Terminated in 50 Ohms . Event triggers must have a minimum pulse width of 1 us and be capable of driving 2 volts into a 50 ohm load.
    - Optical isolation.
    - Green LED indicating the channel has been triggered by a VMEbus or external event trigger. Pulse stretched for visibility.
    - External trigger inputs available on P2 user pins.
- TRIGGER:
  - Green LED, indicating that one or more channels has been triggered by the VMEbus or external event trigger input. Pulse stretched for visibility.
- ERROR:
  - Red LED, indicating that one or more V101 event trigger channels have exceeded their input capacity, and one or more event triggers have not been transmitted. Pulse stretched for visibility. ERROR also indicates a VMEbus interrupt request (if interrupts are enabled).
- OFFLINE:
  - Red LED, indicating that the front end computer has not enabled the module. No event triggers can be processed by the module. However, the module will pass through triggers from active lower priority input modules.
- VME SEL:
  - Green LED, indicates that the input module has been addressed as a VMEbus slave, read or write. Pulse stretched for visibility.

3.3 V101 Event Input Module Characteristics

3.3.1 V101 Event Input Module BASE_ADDRESS

The V100 event encoder module supports up to 16 V101 event input modules. The 16 V101 event input module registers are separately addressed. The input module address is comprised of two parts; BASE_ADDRESS, and MODULE_ADDRESS. The simple decoding used in VMEbus A16 interface requires that the input modules be assigned a block of 0x800-bytes (2048). An V101 event input module address is obtained from the encoder module BASE_ADDRESS, and the module’s position on the 16-position input module bus. The V101 event input module BASE_ADDRESS is:

```
_________________MSB-----------ADDRESS-16------------LSB
________________|15|14|13|12|11|10|09|08|07|06--------00|
BASE_ADDRESS____| X| X| X| X| 0| 0| 0| 0| 0|0----------0|
INPUT_OFFSET____| 0| 0| 0| 0| 1| 0| 0| 0| 0|0----------0|
MODULE_POSITION_| 0| 0| 0| 0| 0| Y| Y| Y| Y|0----------0|
MODULE_ADDRESS__| X| X| X| X| 1| Y| Y| Y| Y|0----------0|
```

Where:

XXXX is the BASE_ADDRESS of the V100 encoder module 0x1000-byte address block
YYYY is the input module event input bus position

NOTE: If an input module is addressed between MODULE_ADDRESS + 0x49 and MODULE_ADDRESS + 0x7F the module will respond, but the results are undefined.

3.3.2 V101 Event Input Module VMEbus Status/ID Message

Located at the input module base address, MODULE_ADDRESS, is the 64-byte, VMEid message. The message contains the module identification information. All even bytes contain an ASCII ".".

3.3.3 V101 Event Input Module Control Registers

The input module control registers are located at MODULE_ADDRESS + 0x40. The V101 event input module control registers are shown below.

The EVNT_CHAN_ENA register (read/write) is a byte addressable, word register, event trigger mask. The 16 V101 event input triggers are bit encoded in this register. Trigger channels are enabled by writing a one into the EVNT_CHAN_ENA register.

NOTE: If the EVNT_CHAN_ENA register is clear, the front panel OFFLINE LED will be illuminated. OFFLINE indicates that the module cannot generate event triggers.

The EVNT_CHAN_INT register (read/write) is a byte addressable, word register, interrupt mask. The 16 V101 event input module channel interrupt on trigger error are bit encoded in this register. Interrupt channels are enabled by writing a one into the EVNT_CHAN_INT register.

The EVNT_CHAN_TRG register (write only) is a byte addressable, word register, event trigger register. The 16 V101 event input module channels may be individually triggered by writing a one into the EVNT_CHAN_TRG register.

The EVNT_CHAN_ERR register (read only) is a byte addressable, word register, channel error register. If a V101 event input module channel has been enabled (EVNT_CHAN_ENA); and the channel is rapidly triggered by the EVNT_CHAN_TRG register (or EXTERNAL_INPUT); the channel bit in the EVNT_CHAN_ERR register will set if a second trigger input is received before the first trigger input has been processed. If the channel bit in the EVNT_CHAN_INT register is set, a V101 event input module interrupt will be generated. The register contents are cleared following a VMEbus read cycle.

The INT_STATUS/ID register (read only) is a byte register interrupt vector. The status I/D byte is obtained from a patch in the V100 encoder module, and the input module's input module bus position. INT_STATUS/ID[7..4] is obtained from the input module bus STAT_ID[7..4]. The input module position, input module bus SWR[10..7], is returned in the interrupt status/ID reply as INT_STATUS/ID[3..0].

The INT_LEVEL register (read/write) is a byte register interrupt request level. The value of INT_LEVEL[3..1] determines the interrupt request/reply level of the V101 event input module. This register must be initialized non-zero, as there is no interrupt request level 0.

	EVEN BYTE								ODD BYTE								
	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
EVNT_CHAN_ENA																	channel enable, bit encoded
EVNT_CHAN_INT																	interrupt enable, bit encoded
EVNT_CHAN_TRG																	event trigger, bit encoded
EVNT_CHAN_ERR																	lost event(s), bit encoded
INT_STATUS/ID									7	6	5	4	3	2	1	0	
INT_LEVEL									0	0	0	0	0	3	2	1	

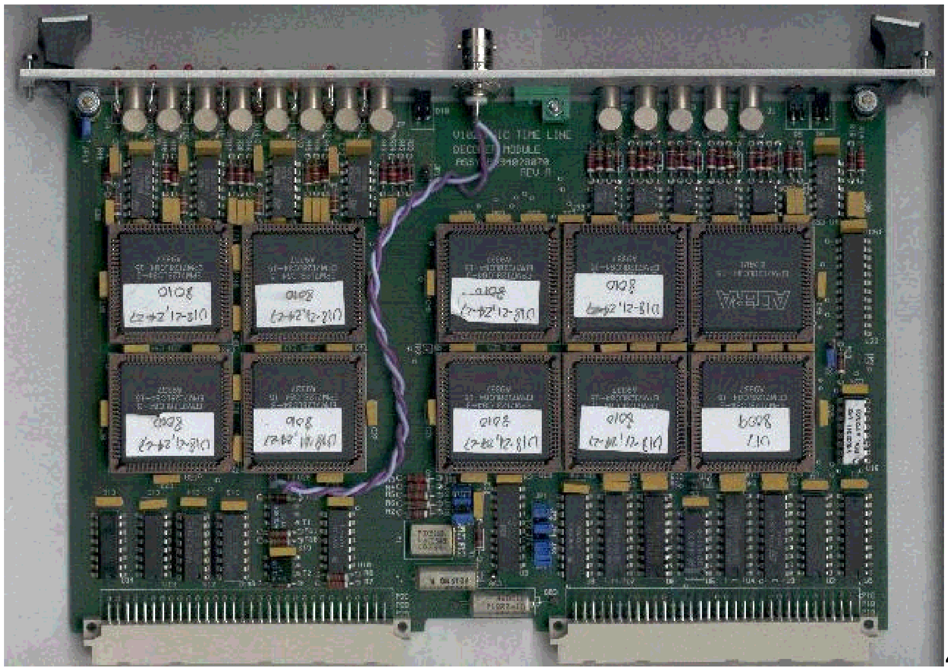
WARNING: There is no requirement for a full complement of 16 V101 event input modules. If V101 event input module is addressed, but not present, a VMEbus time-out will occur (BERR).

3.4 V101 Input Module Jumper Setup Details

There are no jumpers on the V101 input module. All setup is done on the encoder module

4.0 V102 DELAY MODULE

Reference: 94028072 Module Assembly - V102 RHIC Timeline Delay Module



4.1 Introduction

The RHIC timeline V102 delay module is a standard VME 6U module. Each module contains eight event output channels with programmable triggers (RHIC timeline event code, previous event output channel, or external), delay times, and delay clock (1 MHz and external).

The module has provision for a direct connection to the RHIC or AGS/Booster event timelines. The timeline events are transmitted short distances, point-to-point, using a twin-axial cable.

4.2 V102 Delay Module Functions

- Synchronizes to the RHIC timeline and detects the RHIC clock and timeline event codes.
  - Detects the RHIC timeline carrier.
  - Detects RHIC timeline event codes.
  - Each RHIC timeline event code is parity and frame checked. If an error is detected, the event is not processed.
  - Derives a 10 MHz clock from the RHIC timeline carrier.
  - Derives 1 MHz, 100 KHz, 10 KHz, and 1 KHz clocks from the 10 MHz clock.
    - The 1 MHz, 100 KHz, 10 KHz, and 1 KHz clocks may be synchronized to the event code stored in the V102 reset register.
    - The 100 KHz, 10 KHz, and 1 KHz clocks may be held on the event code stored in the V102 reset register, and released on the next channel 1 output pulse.
  - If enabled, generates a VMEbus interrupt on event code parity or framing error, or timeline carrier loss. The status/ID vector, and interrupt request level are programmable.
- Detects specified event codes on the timeline and:
  - Initiates an event channel programmable delay. Note - event channels may be programmed to respond to one or more timeline event codes.
  - The delay is developed in a 32-bit counter, programmed to count down.
    - The minimum delay is 1 count.
    - The maximum delay is 232 counts, 430 seconds at 10 MHz or 4300 seconds (70 minutes) at 1 MHz.
  - The channel delay clock may be selected from:
    - The 10 MHz clock derived from the RHIC timeline.
    - The 1 MHz clock derived from the RHIC timeline.
    - A common external clock. Note - the maximum frequency is 5 MHz.
  - Event channel trigger options are:
    - RHIC timeline event(s).
    - The preceding event channel trigger (an external trigger coupled by an external cable in the case of channel 1).
    - External trigger (channel pairs 1/2, 3/4, 5/6, 7/8).
      VMEbus command. See below for an additional trigger

option.

- At the end of the delay, an event channel output pulse is developed:
  - The pulse width is developed in a 16-bit counter, programmed to count down.
  - The minimum pulse width is one delay clock period.
    - 0.1 usec using the internal 10 MHz clock.
    - 1 usec using the internal 1 MHz clock.
    - 1/f ext using an external clock.
  - The maximum pulse widths are:
    - 6.5 msec using the internal 10 MHz clock.
    - 65 msec using the internal 1 MHz lock.
    - 216 x 1/fext using an external clock.
  - The pulse is capable of driving a 50 ohm load.
    - Minimum pulse level at the event decode module output is +3V.
    - Event channel outputs may be wire ’ored’.
    - Event channel outputs are available on the front panel and rear VMEbus P2 user pins.
    - The pulse sets a read only, bit in the clock/trigger selection register.
    - Event channel pair output pulse set/reset mode.
    - Available on channel pairs 1-2, 2-3, 3-4, 4-5, 5-6, 6-7, 7-8:
    - The leading edge of the first channel (lower channel number) pulse output sets the pulse output of the second channel (higher channel number).
    - The leading edge of the second channel (higher channel number pulse output resets its pulse output.
- RHIC/AGS mode switch (module jumper patch).
  - When set in RHIC mode:
    - The clock outputs are differential TTL square waves with a 50% duty cycle. The pulse outputs are positive single ended.
  - When set to the AGS mode:
    - Outputs drive a 25 wire ribbon cable which mates to the D09-E1819 TTL to AGS pulse converter.
      - The clock outputs are negative going short pulses.
      - The pulse outputs are negative going.
      - Pulse period must be set to 0.1 or 1.0 usec pulse width in the clock/trigger selection register.

All triggers, timeline event, VMEbus, or external are synchronized to the selected clock (external or internal) before the delay count down begins.

Each V102 delay module is completely self supporting, once initialized. There can be more than one V102 delay module in a VME chassis. The module driver software should be capable of supporting a full chassis of 16 V102 delay modules. Each module requires its event code trigger mask table and registers to be initialized for proper operation. Once these items are initialized, no further support is required for normal operation. The items are;

- Event code trigger mask table, 256-bytes.
- Command and status register, 1-byte.
- Master reset event code register, 1-byte.
- Interrupt status/ID (vector), 1-byte.
- Interrupt level, 1- byte.
- 8-channel register sets:
  - delay counter register, 4-bytes.
  - pulse width register, 2-bytes.
  - clock, trigger, and mode register, 1-byte.
  - delay/pulse width counter stop/clear register, 1-byte.

4.3 V102 Delay Module Front Panel

- VME SEL:
  - Green LED, indicates that the delay module has been addressed as a slave, VMEbus read or write. Pulse stretched for visibility.
- EXTERNAL TRIGGER AND CLOCK INPUTS:
  - Four external trigger inputs; pairs 1/2, 3/4, 5/6, 7/8.
    - LEMO NIM-CAMAC connectors
    - Available on P2 user pins.
- Single external clock input; common to all channels.
  - LEMO NIM-CAMAC connector
  - Available on P2 user pin.
- INPUT CONNECTOR:
  - Amphenol 3-2225 twin-axial connector
    - Transformer coupling for galvanic isolation.
    - Available on P2 user pins.
- EVENTLNK:
  - Green LED, indicating timeline carrier detected.
- EVENT CHANNEL OUTPUTS:

- Eight event channels.
- LEMO NIM-CAMAC output connector.
  - Available on P2 user pins.
- Green LED indicating each output pulse. Pulse stretched for visibility.

4.4 V102 Delay Module Characteristics

4.4.1 V102 Delay Module BASE_ADDRESS

The V102 delay module BASE_ADDRESS is obtained from an address jumper patch. The V102 delay module is addressed VMEbus A16. The simple decoding used in the interface requires that the delay modules be assigned a block of 0x200-bytes (512).

```
______________MSB___________ADDRESS-16____________LSB
_____________|15|14|13|12|11|10|09|08--------------00|
BASE_ADDRESS |_X|_X|_X|_X|_X|_X|_X|0----------------0|
```

4.4.2 V10 Delay Module VMEbus Status/ID Message

Located at the delay module BASE_ADDRESS, is the 64-byte, VMEid message. The message contains the module identification information. All even bytes contain an ASCII ".".

4.4.3 V102 Delay Module Control Registers

4.4.3.1 Command and Status Register

The command and status register is located at BASE_ADDRESS + 0x41. The register should be the last register set during V102/V104 module initialization. When GO sets the V102/V104 module will process timeline event codes, and pass the recovered event codes through the event code channel trigger mask table. If an event code channel trigger mask table output contains a one, and the channel is triggered.

NOTE: The front panel OFFLINE red LED indicates the state of the GO bit. If GO is reset, OFFLINE indicates that the delay module is not functional.

The interrupt enable bit (INEN) enables interrupts on timeline carrier failure or parity error. If the interrupts are not enabled the command and status may be read to determine any operating errors. The interrupt status bits are read only, and clear after a read.

The timeline derived clocks; 1 MHz, 100 KHz, 10 KHz, and 1 KHz, are controlled by the enable reset bit (ERST), and the hold bit (HOLD). If ERST is set and HOLD is reset, detection of the reset event code (RESET_EVT_CODE) will reset the timeline clock divider chain. If HOLD is set, ERST becomes a don't care. In this case 100 KHz, 10 KHz, and 1 KHz hold (clock divider held reset) when the reset event code (RESET_EVT_CODE) is detected. The hold is cleared by the next channel 1 event code. As the channel 1 event code clears hold, 1 MHz is reset.

```
BASE_ADDRESS + 0x41| 07---------------------------------00 |
CMD_STA_REG________|INTR|CLCF|__PE|___0|HOLD|ERST|INEN|__GO|
```

where: INTR = (read only)logic OR of bits 06 and 05
CLFL = (read only) timeline failure; no carrier detected
PE = (read only) timeline parity error detected
HOLD = (read/write) hold on RESET_EVT_CODE, release on channel 1
ERST = (read/write) reset the clocks when RESET_EVT_CODE is detected
INEN = (read/write) enable interrupt
GO = (read/write) command start the decode module

Note:
1. The CMD_STA_REG parity bit PE, CLFL, read only, is cleared after a VMEbus read cycle. An interrupt is generated for each parity error (if enabled).
2. The CMD_STA_REG timeline failure bit, CLCF, read only, is cleared by a VMEbus read following timeline restoration. A single interrupt is generated at the beginning of a timeline carrier failure.

4.4.3.2 Reset Event Code Register

Reset event code register is located at BASE_ADDRESS + 0x43. The register contents are compared to each event code received. If equal, the 1 MHz, 100 KHz, and 10 KHz counters are reset or held as described in paragraph 4.4.3.1 above.

```
BASE_ADDRESS + 0x43| 07---------------------------------00 |
RESET_EVT_CODE_____|_R7_|_R6_|_R5_|_R4_|_R3_|_R2_|_R1_|_R0_|
```

NOTE: The event code assigned as the reset event code is processed normally the remainder of the V102 module.

4.4.3.3 Interrupt Status/ID Register

The V102/V104 module interrupt status/ID register (interrupt vector) is located at BASE_ADDRESS + 0x45. The register is read/write. The contents of the status/ID byte are is determined by front end computer interrupt vector requirements. The contents of the register becomes the V102/V104 module interrupt status/ID (interrupt vector) returned during an interrupt cycle.

```
BASE_ADDRESS + 0x45| 07---------------------------------00 |
INTERRUPT_SID______|_I7_|_I6_|_I5_|_I4_|_I3_|_I2_|_I1_|_I0_|
```

4.4.3.4 Interrupt Level

The V102/V104 module interrupt level register is located at BASE_ADDRESS + 0x47. The register is read/write. The contents of the interrupt level register are used to determine the interrupt request level. The valid range of interrupt levels are 7..1. The contents of the interrupt level are compared to VMEbus A[3..1] during an interrupt cycle. If the module is requesting an interrupt, the request level must equal the response level before the module responds.

```
BASE_ADDRESS + 0x47| 07---------------------------------00 |
INTERRUPT_LEV______|___0|___0|___0|___0|___0|_L3_|_L2_|_L1_|
```

4.4.3.4 Event Delay Module Delay, Pulse Width, and Clock/Trigger Registers

The V102 event delay module event output channel delay, pulse width, and clock/trigger selection registers are a 64-byte block starting at BASE_ADDRESS + 0x48 as shown below.

The minimum value that may be assigned to the 32-bit delay register, and 16-bit pulse width register is one. If either register contains zero, the channel is disabled, even if properly triggered.

The counter control register selects the delay/pulse width counter trigger and clock. The counter stop register clears and stops the delay and pulse with counters. If the channel counter stop register STOP bit is set, the counters (delay and pulse width) are immediately cleared, and held clear until the STOP bit is cleared. STOP also clears the trigger-clock synchronizer, and disables the trigger input.

```
BASE_ADDRESS + 0x48| 07---------------------------------00|
DELAY_MSB__________| C31| C30| C29| C28| C27| C26| C25| C24|
DELAY______________| C23| C22| C21| C20| C19| C18| C17| C16|
DELAY______________| C15| C14| C13| C12| C11| C10| C09| C08|
DELAY_LSB__________| C07| C06| C05| C04| C03| C02| C01| C00|
PULSE_MSB__________| P15| P14| P13| P12| P11| P10| P09| P08|
PULSE_LSB__________| P07| P06| P05| P04| P03| P02| P01| P00|
COUNTER_CTRL_______|PULS|FLIP|CLK1|CLK0|TRIG|INVT|TRG1|TRG0|
COUNTER_STOP_______|___0|___0|___0|___0|___0|___0|___0|STOP|
```

where:
COUNTER_CTRL functions are:
PULS; channel has been triggered since previous register read, read only, clear after read.
FLIP; leading edge of the previous counter input sets the pulse output, and the leading edge of the channel output pulse resets the pulse output.
CLK[4..5] = 0 = no clock selected
CLK[4..5] = 1 = select 10 MHz clock
CLK[4..5] = 2 = select 1 MHz clock
CLK[4..5] = 3 = select external clock
TRIG; trigger channel immediately (may be used with other trigger modes)
INVT; invert the output pulse (normally positive true) TRG[1..0] = 0 = no trigger selected
TRG[1..0] = 1 = select timeline event trigger
TRG[1..0] = 2 = select external trigger (channel pairs 1&2, 3&4, 5&6, and 7&8 share triggers)

TRG[1..0] = 3 = select trigger on previous counter output (except channel 0)

V102 Event Delay Module Event Channel Output Register

4.4.3.5 V102 Module Event Code Channel Trigger Mask Registers

The V102 module event code channel trigger mask register is a 256 byte SRAM starting at the module BASE_ADDRESS + 0x100. The mask SRAM address is the received and verified event codes. The SRAM contents are bit encoded, trigger masks. The mask SRAM is addressed, and then its contents are used to mask each output channel's trigger. Output channels with a one in their trigger mask position are triggered.

```
______________|MSB------------------------------- LSB|
EVENT_CODE_00_|CH08|CH07|CH06|CH05|CH04|CH03|CH02|CH01|
"
"
EVENT_CODE_FF_|CH08|CH07|CH06|CH05|CH04|CH03|CH02|CH01|
```

4.5 V102 Delay Module Jumper Setup Details

V102 JUMPERS		
LOCATION	**JUMPER**	**FUNCTION**

JP[1..7]	ON = ZERO
	OFF = ONE
		**VMEbus A16 ADDRESS
		A[15..9]**

V102 CLOCK RECOVERY CIRCUIT		
PRINTED WIRE FIXED OCTAVE 5 N2 N1 N0 L2	**JUMPERS JP8 JP9 JP10 L1 L0 N**	**CENTER FREQUENCY MHz**
0 0 1 0	ON OFF ON	12.2
"	ON OFF OFF	10.1
"	OFF ON ON	9.3
"	OFF ON OFF	8.6
"	OFF OFF ON	8.0
"	OFF OFF OFF	7.5
JP11	ON OFF	FRONT PANEL USER P2 PINS
V102 JUMPERS		
LOCATION	**JUMPER**	**FUNCTION**
JP12	ON OFF	**RHIC MODE AGS MODE**
JP[13..14]	**ON OFF OFF ON**	**PULSE[1..7] REFERENCE DGND AGND**

5.0 V103 RHIC MASTER RESET MODULE

Reference: 94028054 Module Assembly - V103 RHIC Master Reset Module

5.1 INTRODUCTION

The V103 RHIC master reset generator transmits two event codes on the RESETLNK to reset remote VME chassis in the RHIC control system. The RESETLNK is separate from other serial control lines such as the EVENTLNK or RTDL. Reset code sequence consists of a first code which must have its MSB set to one, and a second code which must have its MSB set to zero. The two code reset sequence is transmitted with minimum headway between codes. The resulting 14-bit reset code allows

16K unique RESET codes.

The RESET codes are transmitted to V108 utility modules installed in VME chassis with front end computers. The V108 utility module tests received RESET code parity, two code sequence (MSB’s), and total transmission time. If all these tests pass, the RESET code is compared to a stored value. If the RESET code compares, the V108 utility module pulses the VME chassis VMEbus SYSRESET* signal line. This resets all the VME modules within the VME chassis.

The V103 master reset module is based on a V100 encoder module printed wiring board, 94028043. The primary modifications are made in the Altera EPLDs. No input modules are required to support the master reset module.

The V103 RHIC master reset module transmission is the same as the EVENTLNK transmission. The master reset module continuously transmits bi-phase-mark binary ones. Using the same transmission technique as the EVENTLNK allows the use of standard twin-axial cable, fiber optic cable, and fanout modules.

When commanded by its front end computer, the V103 master reset module transmits a 2-byte RESET code. Each code includes a binary zero start bit, 8-data bits (MSB first), even parity bit, and two stop bits (binary ones). The MSB (first bit) of each RESET code byte is used as a first-second byte indicator. 1.2 usec are required to transmit a code byte (2.4 usec for the RESET code sequence). The RESETLNK transmission line contains continuous bi-phase-mark "ones" transmission during idle periods, providing receivers with a 10 MHz clock.

5.2 V103 RHIC MASTER RESET MODULE CHARACTERISTICS

5.2.1 V103 RHIC Master Reset Module Addressing

The basic V100 encoder module VME A16 addressing is retained in the V103 master reset module. The V100 encoder module requires 0x1000 bytes, and its address patch is 4-bits. The V103 master reset module VME interface only utilizes 0x80-bytes of A16 address space. However, the 94028043 printed circuit board address patch is limited to 4-bits. Therefore, the V103 master reset module VME address must be one of the 16 shown:

0x0000, 0x1000, 0x2000 0x3000 .... or 0xF000

5.2.2 V103 RHIC Master Reset Module Registers

The V103 master reset module VMEbus data interface is D8(OE). The VMEid message utilizes the first 0x40-bytes of the address space. The VMEid even bytes are an ASCII ".", and the odd bytes are the VMEid message.

Three of the next four bytes are the V103 master reset module registers. In the command and status register, address base +0x041, a single ENAble bit controls the reset system. The master reset module cannot transmit RESET codes until ENAble is set. However, it will transmit the idle carrier when not enabled. The next two bytes, BASE_ADDRESS + 0x042 and 0x043 are the RESET code. The RESET code transfer from the front end computer to the reset module requires two byte transfers, not a one word transfer. As the second code byte is loaded, the VMEbus register write strobe starts a state machine controller to transfer the two bytes to the RESET code transmitter. The RESET code registers are VMEbus write only, the results of a VMEbus read are undefined.

VMEbus A16 register locations:

```
|15|14|13|12|11|10|09|08|07|06|05|04|03|02|01|00|
|SWR_PATCH__|_0__0__0__0__0__X__X__X__X__X__X| VME address block
|SWR_PATCH__|_0__0__0__0__0__0__X__X__X__X__X| VME status/ID PROM
|SWR_PATCH__|_0__0__0__0__0__1__X__X__X__0__1| command and status
register
|SWR_PATCH__|_0__0__0__0__0__1__X__X__X__1__0| RESET code first
byte
|SWR_PATCH__|_0__0__0__0__0__1__X__X__X__1__1| RESET code second
byte
```

5.2.3 V103 RHIC Master Reset Module Register Contents

The command and status register, BASE_ADDRESS + 0x041: a single ENAble bit controls the reset system. The master reset module cannot transmit RESET codes until ENAble is set. However, it will transmit the idle carrier when not enabled.

The next two bytes, BASE_ADDRESS + 0x042 and 0x043 are the RESET code. The MSB of each RESET code byte is reserved as a first byte (one), second byte (zero) indicator. The one, and zero is shown below, however they are software controlled. The remaining 14-bits of the RESET code encode VME chassis number. The RESET code transfer from the front end computer to the reset module requires two byte transfers, not a one word transfer. As the second code byte is loaded, the VMEbus register write strobe starts a state machine controller to transfer the two bytes to the RESET code transmitter. The RESET code registers are VMEbus write only, the results of a VMEbus read are undefined.

```
REGISTER______|MSB_____________________________LSB|
CMD_STAT_REG_|_0_|_0_|_0_|_0_|_0_|_0_|_0_|ENA|read/write
FIRST_RESET__|_1_|_X_|_X_|_X_|_X_|_X_|_X_|_X_|write only
SECOND_RESET_|_0_|_X_|_X_|_X_|_X_|_X_|_X_|_X_|write only
```

5.2.4 V103 Master Reset Module Jumper Setup Details

V103 JUMPERS		
LOCATION	**JUMPER**	**FUNCTION**

JP[1..4]	ON = ZERO
	OFF = ONE
		VMEbus A16 ADDRESS A[15..12]

6.0 V104 DECODER MODULE

Reference: 94028175 Module Assembly - V104 RHIC Timeline Decoder Module

6.1 Introduction

The RHIC timeline V104 decoder module is a standard VME 6U module. The V104 decoder module is based on the V102 delay module printed wiring board 94028069. The primary difference is the removal of the Altera delay/pulse width EPLDs. Each module contains eight event output channels with programmable triggers (RHIC timeline event code, previous event output channel, or external), and fixed 1 uses pulse widths. The output pulses are triggered as soon as the channel event code is detected.

6.2 V104 Decoder Module Functions

- Synchronizes to the RHIC timeline and detects the RHIC clock and timeline event codes.
  - Detects the RHIC timeline carrier.
  - Detects RHIC timeline event codes.
  - Each RHIC timeline event code is parity and frame checked. If an error is detected, the event is not processed.
  - Derives a 10 MHz clock from the RHIC timeline carrier.
  - Derives 1 MHz, 100 KHz, 10 KHz, and 1 KHz clocks from the 10 MHz clock.
    - The 1 MHz, 100 KHz, 10 KHz, and 1 KHz clocks may be

        synchronized to the event code stored in the V104 reset register.
    - The 100 KHz, 10 KHz, and 1 KHz clocks may be held on the event code stored in the V104 reset register, and released on the next channel 1 output pulse.
- If enabled, generates a VMEbus interrupt on event code parity or framing error, or timeline carrier loss. The status/ID vector, and interrupt request level are programmable.
- Detects specified event codes on the timeline and initiates a 1 usec output pulse.
    - The pulse is capable of driving a 50 ohm load.
        - Minimum pulse level at the event decode module output is +3V.
        - Event channel outputs may be wire 'ored'.
        - Event channel outputs are available on the front panel and rear VMEbus P2 user pins.
- RHIC/AGS mode switch (module jumper patch).
    - When set in RHIC mode:
        - The clock outputs are differential TTL square waves with a 50% duty cycle. The pulse outputs are positive single ended.
    - When set to the AGS mode:
        - Outputs drive a 25 wire ribbon cable which mates to the D09-E1819 TTL to AGS pulse converter.
            - The clock outputs are negative going short pulses.
            - The pulse outputs are negative going.

Each V104 decoder module is completely self supporting, once initialized. There can be more than one V104 decoder module in a VME chassis. The module driver software should be capable of supporting a full chassis of 16 V104 decoder modules. Each module requires its event code trigger mask table and registers to be initialized for proper operation. Once these items are initialized, no further support is required for normal operation. The items are;

- Event code trigger mask table, 256-bytes.
- Command and status register, 1-byte.
- Master reset event code register, 1-byte.
- Interrupt status/ID (vector), 1-byte.
- Interrupt level, 1- byte.

6.3 V104 Decoder Module Front Panel

- VME SEL:
  - Green LED, indicates that the delay module has been addressed as a slave, VMEbus read or write. Pulse stretched for visibility.
- INPUT CONNECTOR:
  - Amphenol 3-2225 twin-axial connector
    - Transformer coupling for galvanic isolation.
    - Available on P2 user pins.
- EVENTLNK:
  - Green LED, indicating timeline carrier detected.
- EVENT CHANNEL OUTPUTS:
  - Eight event channels.
  - LEMO NIM-CAMAC output connector.
    - Available on P2 user pins.
- Green LED indicating each output pulse. Pulse stretched for visibility.

6.4 V104 Decoder Module Characteristics

6.4.1 V104 Decoder Module BASE_ADDRESS

The V104 Decoder Module BASE_ADDRESS is obtained from an address jumper patch. The V104 Decoder Module is addressed VMEbus A16. The simple decoding used in the interface requires that the delay modules be assigned a block of 0x200-bytes (512).

```
______________MSB___________ADDRESS-16____________LSB
______________|15|14|13|12|11|10|09|08--------------00|
BASE_ADDRESS |_X|_X|_X|_X|_X|_X|_X|0----------------0|
```

6.4.2 V10 Delay Module VMEbus Status/ID Message

Located at the delay module BASE_ADDRESS, is the 64-byte, VMEid message. The message contains the module identification information. All even bytes contain an ASCII ".".

6.4.3 V104 Decoder Module Control Registers

6.4.3.1 Command and Status Register

The command and status register is located at BASE_ADDRESS + 0x41. The register should be the last register set during V102/V104 module initialization. When GO sets the V102/V104 module will process timeline event codes, and pass the recovered event codes through the event code channel trigger mask table. If an event code channel trigger mask table output contains a one, and the channel is triggered.

NOTE: The front panel OFFLINE red LED indicates the state of the GO bit. If GO is reset, OFFLINE indicates that the delay module is not functional.

The interrupt enable bit (INEN) enables interrupts on timeline carrier failure or parity error. If the interrupts are not enabled the command and status may be read to determine any operating errors. The interrupt status bits are read only, and clear after a read.

The timeline derived clocks; 1 MHz, 100 KHz, 10 KHz, and 1 KHz, are controlled by the enable reset bit (ERST), and the hold bit (HOLD). If ERST is set and HOLD is reset, detection of the reset event code (RESET_EVT_CODE) will reset the timeline clock divider chain. If HOLD is set, ERST becomes a don't care. In this case 100 KHz, 10 KHz, and 1 KHz hold (clock divider held reset) when the reset event code (RESET_EVT_CODE) is detected. The hold is cleared by the next channel 1 event code. As the channel 1 event code clears hold, 1 MHz is reset.

```
BASE_ADDRESS + 0x41| 07----------------------------------00 |
CMD_STA_REG________|INTR|CLCF|__PE|___0|HOLD|ERST|INEN|__GO|
where: INTR = (read only)logic OR of bits 06 and 05
CLFL = (read only) timeline failure; no carrier detected
PE = (read only) timeline parity error detected
HOLD = (read/write) hold on RESET_EVT_CODE, release on channel 1
ERST = (read/write) reset the clocks when RESET_EVT_CODE is detected
INEN = (read/write) enable interrupt
GO = (read/write) command start the decode module
```

Note:

1. The CMD_STA_REG parity bit PE, CLFL, read only, is cleared after a VMEbus read cycle. An interrupt is generated for each parity error (if enabled).

2. The CMD_STA_REG timeline failure bit, CLCF, read only, is cleared by a VMEbus read following timeline restoration. A single interrupt is generated at the beginning of a timeline carrier failure.

6.4.3.2 Reset Event Code Register

Reset event code register is located at BASE_ADDRESS + 0x43. The register contents are compared to each event code received. If equal, the 1 MHz, 100 KHz, and 10 KHz counters are reset or held as described in paragraph 6.4.3.1 above.

```
BASE_ADDRESS + 0x43| 07---------------------------------00 |
RESET_EVT_CODE_____|_R7_|_R6_|_R5_|_R4_|_R3_|_R2_|_R1_|_R0_|
```

NOTE: The event code assigned as the reset event code is processed normally the remainder of the V102 module.

6.4.3.3 Interrupt Status/ID Register

The V104 module interrupt status/ID register (interrupt vector) is located at BASE_ADDRESS + 0x45. The register is read/write. The contents of the status/ID byte are is determined by front end computer interrupt vector requirements. The contents of the register becomes the V102/V104 module interrupt status/ID (interrupt vector) returned during an interrupt cycle.

```
BASE_ADDRESS + 0x45| 07---------------------------------00 |
INTERRUPT_SID______|_I7_|_I6_|_I5_|_I4_|_I3_|_I2_|_I1_|_I0_|
```

6.4.3.4 Interrupt Level

The V104 module interrupt level register is located at BASE_ADDRESS + 0x47. The register is read/write. The contents of the interrupt level register are used to determine the interrupt request level. The valid range of interrupt levels are 7..1. The contents of the interrupt level are compared to VMEbus A[3..1] during an interrupt cycle. If the module is requesting an interrupt, the request level must equal the response level before the module responds.

```
BASE_ADDRESS + 0x47| 07---------------------------------00 |
INTERRUPT_LEV______|___0|___0|___0|___0|___0|_L3_|_L2_|_L1_|
```

6.4.3.5 V104 Module Event Code Channel Trigger Mask Registers

The V104 module event code channel trigger mask register is a 256 byte SRAM starting at the module BASE_ADDRESS + 0x100. The mask SRAM address is the received and verified event codes. The SRAM contents are bit encoded, trigger masks. The mask SRAM is addressed, and then its contents are used to mask each output channel’s trigger. Output channels with a one in their trigger mask position are triggered.

```
______________|MSB-------------------------------- LSB|
EVENT_CODE_00_|CH08|CH07|CH06|CH05|CH04|CH03|CH02|CH01|
"
"
EVENT_CODE_FF_|CH08|CH07|CH06|CH05|CH04|CH03|CH02|CH01|
```

6.4.5 V104 Decoder Module Jumper Setup Details

V104 JUMPERS		
LOCATION	**JUMPER**	**FUNCTION**

JP[1..7]	ON = ZERO
	OFF = ONE
		**VMEbus A16 ADDRESS
		A[15..9]**

V104 CLOCK RECOVERY CIRCUIT		
PRINTED WIRE FIXED OCTAVE 5 N2 N1 N0 L2	**JUMPERS JP8 JP9 JP10 L1 L0 N**	**CENTER FREQUENCY MHz**
0 0 1 0	ON OFF ON	12.2
"	ON OFF OFF	10.1
"	OFF ON ON	9.3
"	OFF ON OFF	8.6
"	OFF OFF ON	8.0
"	OFF OFF OFF	7.5
JP11	ON OFF	FRONT PANEL USER P2 PINS
V104 JUMPERS		
LOCATION	**JUMPER**	**FUNCTION**
JP12	ON OFF	**RHIC MODE AGS MODE**
JP[13..14]	**ON OFF OFF ON**	**PULSE[1..7] REFERENCE DGND AGND**

2.8 Multiplexed Analog to Digital Converters

2.8.1 MADC Paper

/home/cfsa/http/htdocs/Hardware/map17.htm

THE RHIC GENERAL PURPOSE MULTIPLEXED ANALOG TO DIGITAL CONVERTER SYSTEM*

R. Michnoff, Brookhaven National Laboratory, Upton, NY 11973 USA

ABSTRACT

A general purpose multiplexed analog to digital converter system is currently under development to support acquisition of analog signals for the Relativistic Heavy Ion Collider (RHIC) at Brookhaven National Laboratory. The system consists of a custom intelligent VME based controller module (V113) and a 14-bit 64 channel multiplexed A/D converter module (V114). The design features two independent scan groups, where one scan group is capable of acquiring 64 channels at 60 Hz, concurrently with the second scan group acquiring data at an aggregate rate of up to 80 k samples/second. An interface to the RHIC serially encoded event line is used to synchronize acquisition. Data is stored in a circular static RAM buffer on the controller module, then transferred to a commercial VMEbus CPU board and higher level workstations for plotting, report generation, analysis and storage.

I. DATA ACQUISITION REQUIREMENTS FOR ACCELERATORS

The most unique requirement for a data acquisition system for accelerators is that sampling of data must be synchronized to accelerator system timing. At the workstation level, data acquired from multiple locations, must be time correlated for plotting and analysis. Another critical requirement for the RHIC system is to provide a historical data buffer, where all data is stored in a continuous circular buffer at a relatively slow acquisition rate (typically 60 Hz) until a critical event such as a beam abort occurs. Selected historical data may then be viewed for analysis, and to determine conditions that caused the critical event. While data is being stored in the historical data buffer, the system must be capable of concurrently scanning a few selected channels at a much faster sampling rate for a snapshot period of time.

II. BACKGROUND

Based on the above requirements and after researching commercially available hardware, it was determined that a custom hardware solution was necessary. Various design options and system architectures were considered, and the finalized design is discussed herein.

III. SYSTEM ARCHITECTURE

A block diagram of the overall system architecture is shown in figure 1, and the front panel layout for each module is shown in figure 2.

The RHIC multiplexed analog to digital converter system also provides the capability to be interfaced with a user supplied sample and hold module. The sample and hold analog outputs connect to the analog input connector on the V114 module. The sample and hold control is generated by the scan trigger output on the V113 module, as shown in figure 3.

A hardware block diagram for the V113 and V114 modules is provided in figure 4. A photograph is also shown. The V113 controller module contains the VME bus interface, RHIC event link interface, data storage memory and all of the data acquisition system real-time intelligence. The V114 converter module contains the analog multiplexers, instrumentation amplifier, A/D converter, and a custom interface to the controller module through user defined pins on the VME bus P2 connector.

IV. SYSTEM FEATURES

A summary of the system’s capabilities is detailed below.

A. Scan Groups

Two idendical scan groups are provided. Each scan group is configured with the following information.

a. Scan list containing all the channels to be scanned.
b. Arm trigger setup.
c. Halt trigger setup.
d. Scan trigger setup.

Typically scan group 0 will contain all defined analog inputs, and will be configured to continuously scan at a rate of 60 Hz for use as a history buffer. Data acquisition will be configured to halt on a specified system event for postmortem analysis of beam aborts. The front end computer reads this buffer to obtain the most recent data for each analog input.

Scan group 1 will normally be used to acquire data according to application specific channel selection and timing requirements.

The following parameters are programmable for each scan group.

Arm/halt trigger source

One source is used for both the arm and halt trigger. This is configurable to be one of the following.

a. Single event or logical OR of group of system events.
b. External trigger.

Scan trigger source

The scan trigger source is configurable to be one of the following.

a. Single event or logical OR of group of system events
b. External trigger.

Arm trigger setup

The arm trigger is defined as the condition that causes data samples to begin being stored in the data buffer. After the arm trigger occurs, data samples will be acquired and stored on every scan trigger until the halt trigger occurs. The arm trigger is configurable to be one of the following.

a. Arm immediately on start command from front end computer
b. Arm on arm/halt trigger source
c. Arm on ms delay after arm/halt trigger source

Scan trigger setup

The scan trigger is defined as the condition that causes a single acquisition of all channels in the scan group. Scan triggering will occur only when scanning is armed as defined by the arm trigger. The scan trigger is configurable to be one of the following.

a. Scan on every period of programmable on-board clock (µs resolution)
b. Scan on scan trigger source.
c. Scan on *n*th scan trigger source
d. Scan on programmable delay after scan trigger source (µs resolution)

Halt trigger setup

The halt trigger is defined as the occurrence of a condition that causes data acquisition to halt. After the halt trigger occurs, data acquistion will stop and an interrupt will be sent to the front end computer. The halt trigger is configurable to be one of the following.

a. Scan continuously until front end computer commands stop
b. Halt when data buffer full
c. Halt on arm/halt trigger source
d. Halt on ms delay after arm/halt trigger source
e. Halt on number of scans after arm/halt trigger source

Reset at end of scan

Each scan group may be configured to reset the arm/halt trigger and the scan enable at the end of each complete scan.

B. On-board Data Buffer Memory

Data buffer memory is typically divided into two buffers, one for the scan group 0 and one for the scan group 1. The data buffer size is programmable in number of full scans. A full scan consists of one reading for each channel in the scan group. Data will be stored in a circular fashion, where the oldest data is overwritten as new data is acquired.

The V113 module may be populated with 1, 2, 3, or 4 Mbytes of data storage memory. The entire data storage memory is mapped to VME A32 space on a switch selectable 4 Mbyte boundary (A31..A22). VME data transfers supported include D32, D16, D08(EO), and BLT (block mode transfer).

Event mask RAM (256 bytes), scan list RAM (256 bytes), program FLASH (128 Kbytes), and configuration registers are mapped to VME A24 space on a switch selectable 256 Kbyte boundary (A23..A18). VME data transfers supported for this area are D08(EO) only.

C. Maximum Scan Rates

Maximum scan rates supported are as follows:

64 chans at 60 Hz for one scan group simultaneously with an aggregate rate of 80 KHz for the second scan group (1chan at 80 KHz, 2 chans at 40 KHz, etc).

Maximum continuous throughput from the V113 controller module to the front end computer, then to the higher level workstation is dependent on the network bandwidth. The goal is to provide a continuous update of 6 channels at 720 Hz to the requesting workstation. This translates to a network bandwidth requirement of:

720 scans/sec * 6 samples/scan * 2 bytes/sample = 8640 bytes/sec

Faster acquisition will be provided with a snapshot mode, where the data acquisition is halted while the front end computer passes data to the higher level workstation.

V. ACKNOWLEDGEMENTS

Thank you to K. Seino, M. Shea, A. Franck, J. Gomilar, and J. Smedinghoff of Fermi National Accelerator Laboratory for describing the architecture, functionality and operational capabilities of the existing and future analog input systems at Fermi Lab during a trip in July 1993. Similar concepts were applied to the design of the BNL RHIC Multiplexed Analog to Digital Converter System.

2.8.2 MADC Specifications

/home/cfsa/http/htdocs/Hardware/madc4htm.htm

Accelerator Development Department RHIC Project

Brookhaven National Laboratory, Associated Universities Inc.

Upton, New York 11973

General Purpose Multiplexed Analog to Digital Conversion System

System Specification

(madc4htm.htm)

Robert Michnoff July 1995

Table of Contents

1.0 Overview

This document describes the RHIC general purpose multiplexed analog to digital converter system. The system consists of a custom intelligent VME based controller module (V113) and a 14-bit 64 channel multiplexed A/D converter module (V114). The design features two independent scan groups, where one scan group is capable of acquiring 64 channels at 60 Hz, concurrently with the second scan group acquiring data at an aggregate rate of up to 80 k samples/second. An interface to the RHIC serially encoded event line is used to synchronize acquisition. Data is stored in a circular static RAM buffer on the controller module, then transferred to a commercial VMEbus CPU board and higher level workstations for plotting, report generation, analysis and storage.

2.0 System Architecture

A generalized software block diagram is provided in figure 0. A block diagram of the overall system architecture is shown in figure 1, and the front panel layout for each module is shown in figure 2. A photograph is also available.

This architecture provides modularity and flexibility. Some possible configurations are as follows.

a. Multiple controller and analog module sets reside in a VME chassis with a single front end computer and additional unrelated VME cards.

b. A single controller and analog module set with a dedicated front end computer housed in a small VME chassis, and mounted in the cabinet where the analog signals are generated. This would minimize analog signal cabling lengths.

2.1 Analog input interface panel

The analog input interface panel provides a general purpose interface to individual analog inputs. This interface panel connects to the multipexer module or the sample and hold module. Input signal connectors are currently undefined. They may be lemo connectors, twinax connectors, coax connectors, or screw terminals. It is also possible that multiple signals generated from a single external enclosure will bypass the analog input interface panel and be connected directly to the multiplexer module or sample and hold module front panel.

2.2 Sample and Hold module (future)

The sample & hold module will provide 16 analog input channels. The sample and hold analog output connector connects to the analog input connector on the multiplexed analog to digital converter module. The sample and hold control is generated by the scan trigger output on the controller module.

The interconnecting diagram for sample and hold analog inputs is shown in figure 3.

2.3 Multiplexed Analog to Digital Converter module

The multiplexed analog to digital converter module provides a 14 bit analog to digital converter and accomodates 64 differential analog input channels. The interface and complete details for this module are defined in specification number ACS/RHIC/94-1922-001.

2.4 Controller module

The controller module contains all of the data acquisition system real-time intelligence. Capabilities for this module are detailed in section 4.

3.0 Hardware Block Diagram

A hardware block diagram for the multiplexed analog to digital converter module and the controller module is provided in figure 4.

4.0 CONTROLLER MODULE

Capabilities of the controller module are detailed in this section.

4.1 Scan Groups

Two idendical scan groups are provided. Each scan group is configured with the following information.

a. Scan list containing all the channels to be scanned.
b. Arm trigger setup.
c. Halt trigger setup.
d. Scan trigger setup.

Details on the configuration parameters are provide in section 4.7.

Typically scan group 0 will contain all defined analog inputs, and will be configured to continuously scan at a rate of 60 Hz for use as a history buffer. Data acquisition will be configured to halt on a specified system event for postmortem analysis of beam aborts. The front end computer can read this buffer to obtain the most recent data for each analog input.

Scan group 1 will normally be used to acquire data for a small number of channels in short bursts of time.

4.2 On Board Memory

Data memory is typically divided into two buffers, one for the scan group 0 and one for the scan group 1. The data buffer size is programmable in number of full scans. A full scan consists of one reading for each channel in the scan group. Data will be stored in a circular fashion, where the oldest data is overwritten as new data is acquired. When the halt trigger setup option is set to "halt when data buffer full" data will not be stored in a circular fashion.

Total data buffer memory size is 4 Mbytes. The entire data buffer memory is mapped to VME A32 space on a switch selectable 4 Mbyte boundary (A31..A22). VME data transfers supported include D32, D16, D08(EO), and BLT (block mode transfer).

Event mask RAM (256 bytes), scan list RAM (256 bytes), program EEPROM (64 Kbytes), and configuration registers are mapped to VME A24 space on a switch selectable 256 Kbyte boundary (A23..A18). VME data transfers supported for this area are D08(EO) only.

4.3 Maximum Scan Rates

Maximum scan rates supported will be at least the following:

64 chans at 60 Hz for one scan group simultaneously with an aggregate rate of 80 KHz for the second scan group (1chan at 80 KHz, 2 chans at 40 KHz, etc).

Maximum continuous throughput from the MADC Controller module to the front end computer, then to the higher level workstation is dependent on the ethernet network bandwidth. The goal is to provide a continuous update of 6 channels at 720 Hz to the requesting workstation. This translates to a network bandwidth requirement of:

720 scans/sec * 6 samples/scan * 2 bytes/sample = 8640 bytes/sec

Faster acquisition will be provided with a snapshot mode, where the data acquisition is halted while the front end computer passes data to the higher level workstation.

4.4 Memory Consumption

The following table provides calculations of data buffer sample periods at different scan frequencies. The calculations are based on each scan containing one 4 byte time stamp, and 4 bytes for each data value.

Scan Frequency	Number of chans in list	time period stored in 2 Mbyte data buffer
60 Hz	64 chans	120 seconds
720 Hz	64 chans	10 seconds
80 KHz	1 chan	3 seconds

Since the number of scans per buffer is programmable, it is possible to dedicate all data memory to one scan group to increase the stored time period. Also, other data storage formats will be available to increase the storage time period. For example, if data only is stored (2 bytes per data value), then a 2 Mbyte data buffer can hold 12 seconds of a single channel at 80 KHz.

4.5 System Configuration Parameters

The following system configuration parameters are programmable for the controller module. These parameters are common for both scan groups. The associated hardware registers are listed for each parameter. See section 4.11 for specific register definitions.

4.5.1 Scan group priority

The scan group priority is programmable to be one of the following.

a. If both scan groups are active, scanning will alternate on each data conversion.
b. Scan group 1 has highest priority. Scan group 0 will be scanned only if scan group 1 is not active.

hardware registers: 0x26003, bit 6

4.5.2 Time stamp clock source select

The time stamp clock source is programmable to be one of the following.

a. Board generated 1 microsecond clock. This is either an on-board 1 microsecond clock or a 1 microsecond clock derived from the event link. See bit 7 in register 0x26007.
b. Event bit number 6. When an event code occurs that has a logic one in bit 6 of the respective event mask ram location, the time stamp is incremented by 1.
hardware registers: 0x26003, bit 7

event mask ram: bit 6 in memory locations 0x20000 through 0x200ff

4.5.3 Board generated 1 microsecond clock select

The board generated 1 microsecond clock is programmable to be one of the following.

This 1 µs clock is used for the time stamp circuit and the delay trigger circuit. The on-board clock will automatically be used if the event link carrier error exists.

a. On-board 1 microsecond clock
b. 1 microsecond clock derived from event link.

hardware registers: 0x26007, bit 7

4.5.4 Multiplexer settling time

The multiplexer settling time is programmable from 0 to 15 microseconds. After the multiplexer channel is changed, the scan sequencer will wait for the programmed settling time before issuing the A/D converter start conversion pulse..

hardware register: 0x26007, bits 0 - 5

4.5.5 Voltage reference select

Voltage references are provided for analog input channels 62 and 63. The voltage reference value for each channel is individually selected via jumpers on the Multiplexed Analog to Digital Converter Board. One control bit is used for both channels 62 and 63, to select one of the following.

a. The actual external analog inputs for channels 62 and 63
b. The jumper selected reference voltage for channels 62 and 63.

hardware register: 0x26007, bit 6

4.5.6 VME Interrupt configuration

A single VME interrupt is generated by the controller module. The VME interrupt level and interrupt vector are programmable via on-board registers, and the interrupt is programmable to be generated by one or more of the following seven conditions.

a. VME command complete
b. sync event/data ready
c. CPU fail
d. A/D error (no response or data FIFO full)
e. Event link carrier error
f. Event link frame error
g. Event link parity error

hardware registers:
0x2a003 - interrupt enable
0x2a005 - interrupt vector
0x2a007 - interrupt level

4.5.7 FLASH code update

The controller module allows the on-board i960 program to be downloaded by VME to the on-board FLASH memory. Before writing to the FLASH memory, the code update must be enabled by writing a special code is to register 0x2a009.

hardware registers: 0x2a009

4.5.8 Reboot command

The controller module provides a feature for VME to reboot the board. A special code is written to register 0x2a009 to perform this operation.

hardware registers: 0x2a009

4.6 System Status Parameters

The following system configuration parameters are provided for the controller module. These parameters are common for both scan groups. The associated hardware registers are listed for each parameter. See section 4.11 for specific register definitions.

4.6.1 Voltage input range status

The voltage input range for all 64 analog input channels is selected via jumpers on the Multiplexed Analog to Digital Converter board. Status bits are provided to read the jumper selected range.

hardware register: 0x26002, bits 6 - 7

4.6.2 Interrupt/Error status

When the VME interrupt occurs, the interrupt status register is read by the interrupt handler to determine the cause of the interrupt. The error status register may be read outside the interrupt handler to determine the current status of the interrupt bits.

hardware registers:
0x2a011 - interrupt status
0x2a001 - error status

4.6.3 A32 base address

The A32 base address dip switch setting may be read by VME to determine the mapping location of the 4 Mbyte memory area.

hardware registers:
0x2a00f
0x2a009, bits 6-7

4.6.4 Memory module status

The controller module contains four memroy module sockets for the A32 memory area. Typically, four 1 MByte memory modules will be installed. However, it is possible to populate the board with only one, two or three modules. A status word is provided to determine the exact memory module population.

hardware register: 0x2a00b

4.6.5 FIFO status

The following FIFO status bits are provided.

a. Data FIFO empty
b. Data FIFO full
c. Data FIFO half full
d. Event FIFO empty
e. Event FIFO full

hardware register:
0x26002
0x24002

4.7 Data Buffer Configuration Parameters

The following data buffer parameters are programmable for each scan group. The associated hardware registers and setup structure elements are listed for each parameter. See the include file "madc2.h" for the definition of the buff_setup structure.

4.7.1 Arm/halt trigger source

One source is used for both the arm and halt trigger. This is configurable to be one of the following.

a. Single event or logical OR of group of system events
b. External trigger.

hardware registers:
0x24000 - trigger source select
0x24001 - external trigger select

4.7.2 Scan trigger source

The scan trigger source is configurable to be one of the following.

a. Single event or logical OR of group of system events
b. External trigger.

hardware registers:
0x24000 - trigger source select
0x24001 - external trigger select

4.7.3 Arm trigger setup

The arm trigger is defined as the condition that causes data samples to begin being stored in the data buffer. After the arm trigger occurs, data samples will be acquired and stored on every scan trigger until the halt trigger occurs. The arm trigger is configurable to be one of the following.

a. Arm immediately on start command from front end computer
b. Arm on arm/halt trigger source
c. Arm on ms delay after arm/halt trigger source

buff_setup structure:
arm_setup
arm_delay

4.7.4 Scan trigger setup

The scan trigger is defined as the condition that causes a single acquisition of all channels in the scan group. Scan triggering will occur only when scanning is armed as defined by the arm trigger. The scan trigger is configurable to be one of the following.

a. Scan on every period of programmable on-board clock (us resolution)
b. Scan on scan trigger source.
c. Scan on *n*th scan trigger source
d. Scan on programmable delay after scan trigger source (us resolution)

hardware registers:
0x26000 - trigger enable
0x28002 - delay trigger configuration
0x28000, 0x28001 - on-board clock count
0x28002 - on-board clock source select

4.7.5 Halt trigger setup

The halt trigger is defined as the occurrence of a condition that causes data acquisition to halt. After the halt trigger occurs, data acquistion will stop and an interrupt will be sent to the front end computer. The halt trigger is configurable to be one of the following.

a. Scan continuously until front end computer commands stop
b. Halt when data buffer full
c. Halt on arm/halt trigger source
d. Halt on ms delay after arm/halt trigger source
e. Halt on number of scans after arm/halt trigger source

buff_setup structure:
halt_setup
halt_delay

4.7.6 Reset at end of scan

Each scan group may be configured to reset the arm/halt trigger and the scan enable at the end of each complete scan.

hardware register: 0x26003, bits 0-1

4.7.7 Front End Computer data ready interrupt mode

The front end computer data ready interrupt mode is configurable to be one of the following.

a. No data ready interrupt

b. Interrupt when data acquisition halts
c. Interrupt on every *n*th scan, and when data acquisition halts

buff_setup structure:
interrupt_mode
int_scan_count

4.7.8 Data buffer size

The data buffer size is programmable in number of scans. The front end computer is responsible for insuring that the required amount of memory is available. The number of bytes in each scan is dependent on the number of channels in the scan list and the selected data storage format.

buff_setup structure: buff_size

4.7.9 Data buffer top pointer

The top of the data buffer is selected by the front end computer. This value is an offset in number of bytes from the top of the A32 memory area, and must reside on an even four-byte boundary.

buff_setup structure: buff_top_ptr

4.7.10 Data storage format

Section 4.9 of this document defines the data storage format options.

buff_setup structure: data_storage_format

4.8 Data Buffer Status Parameters

The following data buffer status parameters are provided for each scan group. The associated status structure elements are listed for each parameter. See the include file "madc2.h" for the definition of the buff_setup structure.

4.8.1 Current data buffer pointer

This is the scan number within the data buffer where the current data scan will be stored.

Note that this value will never be greater than buff_size as defined in the buff_setup structure.

buff_status structure: current_buff_ptr

4.8.2 Scan count since last sync event

This is a count of the number of scans since the last sync event occurred.

buff_status structure: current_buff_ptr

4.8.3 Buffer pointer at time of last sync event

This is the scan number within the data buffer at the time of the last sync event.

This is used for data syncronization across multiple boards.

buff_status structure: buff_ptr_at_time_of_last_sync

4.8.4 Scan count at time of last sync event

This is the count of the number of scans at the time of the last sync event.

This is used for data syncronization across multiple boards.

buff_status structure: scan_count_at_time_of_last_sync

4.8.5 Status flags

Individual bits in this parameter are used as status flags. The following status conditions are available.

BUFFER_ACTIVE
: This bit indicates that scanning is in progress, or that the buffer is waiting for an arm condition to occur.

BUFFER_FULL
: This bit indicates that the data buffer is full. This is also used to indicate that data storage has wrapped around in the circular data buffer.

WAITING_FOR_ARM_TRIG
: This bit indicates that the buffer is waiting for an arm trigger to occur.

WAITING_FOR_HALT_TRIG

This bit indicates that the buffer is waiting for a halt trigger to occur.

WAITING_FOR_ARM_DELAY

This bit indicates that the arm trigger has occurred and the buffer is waiting for millisecond arm time delay to expire.

WAITING_FOR_MS_HALT_DELAY

This bit indicates that the halt trigger has occurred and the buffer is waiting for the millisecond halt time delay to expire.

WAITING_FOR_SCAN_HALT_DELAY

This bit indicates that the halt trigger has occurred and the buffer is waiting for the scan count halt time delay to expire.

buff_status structure: status_flags

4.8.6 Arm delay count

This is the current value of the arm delay counter.

buff_status structure: arm_delay

4.8.7 Halt delay count

This is the current value of the halt delay counter.

buff_status structure: halt_delay

4.8.8 Interrupt scan count

This is the current value of the interrupt scan counter.

buff_status structure: int_scan_count

4.8.9 Bad scan status

Three parameters are used to define the current status of bad scans. since the last sync event. bad_scan_count is a count of the number of bad scans in the data buffer since scanning began. The i960 program counts the number of channels in the list for each scan. When the last channel is detected (the most significant bit in the channel is a logic 1), the i960 program compares the count of the number of channels in the list with the number defined by num_of_chans_in_list in the buff_setup structure. When the numbers are not equal, bad_scan_count is incremented. first_bad_scan is the scan number of the first bad scan. last_bad_scan is the scan number of the most recent bad scan.

buff_status structure: bad_scan_count

4.9 Data storage format

The system will be configurable to store data with or without a time tag, and with or without a channel number tag. The possible data storage format options are diagrammed below.

Option 0

Option 1

4.10 VME Memory Map

A32 address space 4 Mbyte boundary Boundary is selected via dip switches for vmea[31..22]			
Address range	**Description**	**Valid address / data modes**	**Size**
000000-3fffff	**Data buffer area**	**09,0d,0b,0f / D08(OE),D16, D32**	**4 Mbytes**

A24 address space 256 Kbyte boundary Boundary is selected via dip switches for vmea[23..18]			
Address range	**Description**	**Valid address / data modes**	**Size (actual used)**
00000-1ffff	FLASH memory	39,3d / D08(OE)	128 Kbytes
00000-0003f	IDPROM	39,3d / D08(O)	64 bytes (32 odd bytes)
20000-21fff	Event mask RAM	39,3d / D08(OE)	8 Kbytes (256 bytes)
22000-23fff	Scan list RAM	39,3d / D08(OE)	8 Kbytes (256 bytes)
24000-25fff	Event config registers	39,3d / D08(OE)	8 Kbytes (3 bytes)
26000-27fff	Scan list config registers	39,3d / D08(OE)	8 Kbytes (4 bytes)
28000-29fff	Delay trigger config registers	39,3d / D08(OE)	8 Kbytes (8 bytes)
2a000-2bfff	VME interface registers	39,3d / D08(O)	8 Kbytes (7 odd bytes)
2c000-2dfff	unused	--	8 Kbytes
2e000-2ffff	unused	--	8 Kbytes
30000-3ffff	unused	--	64 Kbytes

Address modes	
09	Extended (32-bit address) nonprivileged data access
0d	Extended (32-bit address) supervisory data access
0b	Extended (32-bit address) nonprivileged block transfer
0f	Extended (32-bit address) supervisory block transfer
39	Standard (24-bit address) nonprivileged data access
3d	Standard (24-bit address) supervisory data access

Data modes	
D08(O)	Single byte access, odd addresses only
D08(OE)	Single byte access, odd or even addresses
D16	Double byte access
D32	32-bit transfer

The MADC controller module does not support uneven double byte access (bytes 1-2).

The MADC controller module does not support triple byte access (bytes 0-2 or bytes 1-3).

4.11 VME Memory and Register Definitions

4.11.1 FLASH memory

The FLASH memory is used to store the on-board 80960 CPU program. The program in FLASH memory may be downloaded through the VME interface. The FLASH memory is normally read only, and is write enabled by writing a special code to the control command register (address 0x2a009).

4.11.2 IDPROM

The IDPROM overlays the first 32 odd bytes of the FLASH memory. The VME ID is a total of 64 bytes long. The even bytes are read from the FLASH memory. The format for the VME ID is the standard format that has been established for RHIC VME boards.

4.11.3 Event mask RAM

The least significant byte of the address in the event mask RAM represents each of the 256 possible event codes. Each of the 8 data bits in each event code address is used to define that a trigger pulse should or should not occur on the respective trigger signals. A bit value of 1 indicates that a trigger pulse should be generated, and a bit value of 0 indicates that a trigger pulse should not be generated. The following list defines the function of each bit in the event mask RAM.

Event Mask RAM Bit Definitions	
Bit number	**Function**
0	scan trigger for scan group 0
1	arm/halt trigger for scan group 0
2	scan trigger for scan group 1
3	arm/halt trigger for scan group 1
4	send event code to fifo
5	sync event, reset timestamp
6	timestamp clock event
7	delay timer clock event

4.11.4 Scan list RAM

The scan list RAM is divided in half. Address locations 0x22000 through 0x22007f contain the scan list for scan group 0, and address locations 0x220080 through 0x2200ff contain the scan list for scan group 1. The scan list fomat is diagrammed.

4.11.5 Event configuration registers

24000: rd/wr trigger config0 register - source select
- bits 1,0 - scan trigger 0 config
- bits 3,2 - halt/arm trigger 0 config
- bits 5,4 - scan trigger 1 config
- bits 7,6 - halt/arm trigger 1 config
  - 0 0 - trigger disabled
  - 0 1 - trigger on event only
  - 1 0 - trigger on external trigger only
  - 1 1 - trigger on external trigger OR event

--
24001: rd/wr trigger config1 register - external trigger select
    bits 1,0 - scan trigger 0 config
    bits 3,2 - halt/arm trigger 0 config
    bits 5,4 - scan trigger 0 config
    bits 7,6 - halt/arm trigger 1 config
        0 0 - select external trigger 0
        0 1 - select external trigger 1
        1 0 - select external trigger 2
        1 1 - select external trigger 3
--
24002: wr simulated event code
--
24002: rd status bits
    bit 0 - event fifo empty
    bit 1 - event fifo full
    bit 2 - event link no carrier error
    bit 3 - event link parity error
    bit 4 - event link frame error
    bit 5-7 - unused

4.11.6 Scan list configuration registers

26000: wr Trigger enable (not normally written to by VME)
    Writing a bit value of 1 will enable the trigger, and a value of 0 will do nothing.
    bit 0 - enable group 0 cascade scan trigger enable
    bit 1 - enable group 1 cascade scan trigger enable
    When the cascade trigger is enabled, the arm/halt event will enable the scan trigger.
    bit 2 - enable group 0 scan trigger
    bit 3 - enable group 1 scan trigger
    bit 4 - enable group 0 arm/halt trigger
    bit 5 - enable group 1 arm/halt trigger
--
26001: wr Trigger disable (not normally written to by VME)
    Writing a bit value of 1 will disable the trigger, and a value of 0 will do nothing.
    bit 0 - disable group 0 cascade scan trigger enable
    bit 1 - disable group 1 cascade scan trigger enable
    bit 2 - disable group 0 scan trigger
    bit 3 - disable group 1 scan trigger
    bit 4 - disable group 0 arm/halt trigger
    bit 5 - disable group 1 arm/halt trigger

--
26003: rd/wr Config 0 register
- bit 0 - when set to 1, group 0 arm/halt trigger will be reset at end of scan
- bit 1 - when set to 1, group 1 arm/halt trigger will be reset end of scan
- bits 3,2 (scan group 0 time stamp control - FUTURE)
- bits 5,4 (scan group 0 time stamp control - FUTURE)
  - 00 - do not store time tag
  - 01 - store time tag with every data point
  - 10 - store time tag on scan trigger
  - 11 - store time tag on arm/halt trigger
- bit 6 - priority
  - 0 - If both scan groups are active, scanning will alternate on each data conversion
  - 1 - Scan group 1 has highest priority. Scan group 0 will be scanned only if scan group 1 is not active.
- bit 7 - time stamp clock source select
  - 0 - time stamp clock source is on-board 1µs clock
  - 1 - time stamp clock source is event #6

--
26007: rd/wr Config 1 register
- bits 5..0 - analog multiplexer settling time
  - 1 count = 16*pclk = 500 ns
  - programmable time = 0 to 31.5 us
- bit 6 - voltage reference select
  - 0 - select external inputs for chans 62 and 63
  - 1 - select jumper selected reference voltage for chans 62 and 63
- bit 7 - 1 µs clock select
  - This 1 µs clock is used for the time stamp circuit and the delay trigger circuit. On-board clock will automatically be used if event link carrier error exists.
  - 0 - select on-board clock
  - 1 - select clock derived from event link

--
26000: rd Trigger enabled/disabled status.
- 0 - trigger is disabled
- 1 - trigger is enabled
- bit 0 - group 0 cascade scan trigger
- bit 1 - group 1 cascade scan trigger
  - When the cascade trigger is enabled, the arm/halt event will enable the scan trigger.
- bit 2 - group 0 scan trigger
- bit 3 - group 1 scan trigger

bit 4 - group 0 arm/halt trigger
bit 5 - group 1 arm/halt trigger

--
26001: rd Trigger status.
0 - trigger is not active
1- trigger is active
bit 0 - group 0 scan trigger
bit 1 - group 1 scan trigger
bit 2 - group 0 arm/halt trigger
bit 3 - group 1 arm/halt trigger
--
26002: rd Status bits.
bit 0 - set (1) indicates data fifo empty
bit 1 - set (1) indicates data fifo full
bit 2 - set (1) indicates data fifo half full
bit 3 - set (1) indicates event fifo empty
bit 4 - set (1) indicates event fifo full
bit 5 - set (1) indicates A/D converter error
End of convert pulse was not detected within timeout period.
bit 7,6 - A/D converter selected voltage input range
Range is selected via jumpers on the A/D converter board.
These bits indicate the selected range.
00 -10 VDC to + 10 VDC
01 -5 VDC to +5 VDC
10 0 VDC to +10 VDC
11 Unused (Also indicates that A/D converter board is probably not connected)

4.11.7 Delay trigger configuration registers

28000: rd/wr Group 0 scan trigger delay counts lsbyte
28001: rd/wr Group 0 scan trigger delay counts msbyte
--
28002: rd/wr Group 0 control bits
bits 2,1,0 -> clock source select
000 - direct clock input (1us)
001 - clock input /10 (10us)
010 - clock input /100 (100us)
011 - clock input /1000 (1ms)
100 - clock input /10000 (10ms)

101 - count delay timer clock events (event #7)
110 - count scan trigger pulses
111 - none

bit 3
0 - count down begins on scan trigger.
pulse out occurs after programmed delay counts.
1 - continuous rate generator.
pulse out occurs after programmed delay counts, and count down automatically restarts

bit 4
0 - pulse out is delayed.
1 - pulse out follows scan trigger.

bit 5
0 - sync event will not reset counter.
1 - if rate generator is selected, sync event will reset counter.

--
28003: rd Group 0 lsbyte of count down value.
28003: wr unused
--
28004: rd/wr Group 1 scan trigger delay counts lsbyte
28005: rd/wr Group 1 scan trigger delay counts msbyte
--
28006: rd/wr Group 1 control bits
bit definition is same as for Group 0 control bits (register 28002)
--
28007: rd Group 1 lsbyte of count down value.
28007: wr unused

4.11.8 VME interface registers

2a011: rd Interrupt status
This register indicates the cause of an interrupt.
When this register is read, the interrupt request will be cleared. However, the interrupt status bit will remain in the logic 1 state until the error condition is cleared.
bit 0 - VME command complete interrupt
bit 1 - sync event/data_ready interrupt
bit 2 - CPU fail interrupt
bit 3 - A/D error (no response) interrupt
bit 4 - Event link carrier error interrupt
bit 5 - Event link frame error interrupt
bit 6 - Event link parity error interrupt

--
2a001: rd Error status

The bits in this register are identical to the interrupt status register, addr 2a011.
Reading this register will NOT cause the interrupt to be cleared.
The front end computer may poll this register after an interrupt occurs, to determine if the error condition still exists.

--
2a003: rd/wr Interrupt enable

The interrupt is enabled when the corresponding bit is high.
bit 0 - VME command complete interrupt
bit 1 - sync event/data_ready interrupt
bit 2 - CPU fail interrupt
bit 3 - A/D error (no response) interrupt
bit 4 - Event link carrier error interrupt
bit 5 - Event link frame error interrupt
bit 6 - Event link parity error interrupt

--
2a005: rd/wr Interrupt vector

bit 0-7 - VME interrupt vector

--
2a007: wr VME Interrupt level

bit 0-2 - interrupt level that the MADC controller module uses to interrupt VME bus.

--
2a007: rd Interrupt level and status bits

bit 0-2 - interrupt level
bit 3 - eeprom code-update
bit 4 - cmd boot
bit 5 - sys fail 0=FAIL, 1=NORMAL
bit 6 - a32sel22 - dip switch setting for address line a22 in a32 base address
bit 7 - a32sel23 - dip switch setting for address line a23 in a32 base address

--
2a009: wr Control command register.

eeprom code-update enable - To enable eeprom code update, write abH, then 9aH to this register.
To disable eeprom code update, write any 8-bit value, excluding abH, 9aH, and 39H.
-
Reboot command - To reboot the MADC controller module, write abH, then 39H to this register.

--
2a009: rd Interrupt level and status bits

bit 0-2 - interrupt level
bit 3 - eeprom code-update
bit 4 - cmd boot
bit 5 - sys fail
bit 6 - a32sel22 - dip switch setting for address line a22 in a32 base address
bit 7 - a32sel23 - dip switch setting for address line a23 in a32 base address

--

2a00b: rd only Memory module PD1, PD0 error.
Bus error will not ocur on write attempt.
bits 1,0 - first memory module (lowest address)
bits 3,2 - second memory module
bits 5,4 - third memory module
bits 7,6 - fourth memory module (highest address)
0 0 - 256k by 32 memory module installed
0 1 - 64k by 32 memory module installed
1 0 - 16k by 32 memory module installed
1 1 - no memory module installed

--

2a00d: wr only Write to this register will trigger interrupt to cpu.

--

2a00f: rd only a32 base address
dip switch setting for address lines a31..a24 in a32 base address
bits 7..0 = a32sel31..a32sel24

4.12 On-Board i960 CPU Memory Map

i960 CPU address space			
Address range	**Description**	**Size**	**Region Width**
5000,0000-5000,07ff	Non-volatile RAM	2 Kbytes (2K x 8)	8 bits
6000,0000-6000,0002	VME Interface Control	3 bytes (3 x 8)	8 bits
7000,0000-7000,0007	Scan list config registers	8 bytes (8 x 8)	8 bits
8000,0000-8000,0000	Data FIFO	4 bytes (1 x 32)	32 bits
9000,0000-9000,0000	Event FIFO	1 byte (1 x 8)	8 bits
a000,0000-a001,ffff	Private RAM	128 Kbytes (32K x 32)	32 bits
b000,0000-b03f,ffff	Shared RAM	4 Mbytes (1M x 32)	32 bits
f000,0000-f001,ffff	FLASH program storage	128 Kbytes (128K x 8)	8 bits

4.12.1 VME Interface Control (write only)

6000,0000: Send interrupt to VME to indicated that VME command is complete
6000,0001: Send interrupt to VME to indicate one of the following:
    1. Sync event occurred
    2. Data is ready to be transferred to VME
6000,0002: Clear SYSFAIL bit. This should be done after cpu initialization is complete.
    If CPU FAIL is asserted, SYSFAIL will not clear.

4.12.2 Scan List Configuration Registers

7000,0000: wr Trigger enable.
    Writing a bit value of 1 will enable the trigger, and a value of 0 will do nothing.
    bit 0 - enable group 0 cascade scan trigger enable
    bit 1 - enable group 1 cascade scan trigger enable
        When the cascade trigger is enabled, the arm/halt event will enable the scan trigger.
    bit 2 - enable group 0 scan trigger
    bit 3 - enable group 1 scan trigger
    bit 4 - enable group 0 arm/halt trigger
    bit 5 - enable group 1 arm/halt trigger
--
70000,0001: wr Trigger disable.
    Writing a bit value of 1 will disable the trigger, and a value of 0 will do nothing.
    bit 0 - disable group 0 cascade scan trigger enable
    bit 1 - disable group 1 cascade scan trigger enable
    bit 2 - disable group 0 scan trigger

bit 3 - disable group 1 scan trigger
bit 4 - disable group 0 arm/halt trigger
bit 5 - disable group 1 arm/halt trigger

--

7000,0002: wr Trigger reset.
Writing a bit value of 1 will reset the trigger, and a value of 0 will do nothing.
bit 0 - reset group 0 scan trigger
bit 1 - reset group 1 scan trigger
bit 2 - reset group 0 arm/halt trigger
bit 3 - reset group 1 arm/halt trigger

--

7000,0000: rd Trigger enabled/disabled status.
0 - trigger is disabled
1- trigger is enabled
bit 0 - group 0 cascade scan trigger
bit 1 - group 1 cascade scan trigger
When the cascade trigger is enabled, the arm/halt event will enable the scan trigger.
bit 2 - group 0 scan trigger
bit 3 - group 1 scan trigger
bit 4 - group 0 arm/halt trigger
bit 5 - group 1 arm/halt trigger

--

7000,0001: rd Trigger status.
0 - trigger is not active
1- trigger is active
bit 0 - group 0 scan trigger
bit 1 - group 1 scan trigger
bit 2 - group 0 arm/halt trigger
bit 3 - group 1 arm/halt trigger

--

7000,0002: rd Status bits.
bit 0 - set (1) indicates data fifo empty
bit 1 - set (1) indicates data fifo full
bit 2 - set (1) indicates data fifo half full
bit 3 - set (1) indicates event fifo empty
bit 4 - set (1) indicates event fifo full

--

7000,0003: rd only Config 0 register
bit 0 - when set to 1, group 0 arm/halt trigger will be reset at end of scan
bit 1 - when set to 1, group 1 arm/halt trigger will be reset end of scan

bits 3,2 (scan group 0 time stamp control - FUTURE)
bits 5,4 (scan group 0 time stamp control - FUTURE)
00 - do not store time tag
01 - store time tag with every data point
10 - store time tag on scan trigger
11 - store time tag on arm/halt trigger
bit 6 - priority
0 - If both scan groups are active, scanning will alternate on each data conversion
1 - Scan group 1 has highest priority. Scan group 0 will be scanned only if scan group 1 is not active.
bit 7 - time stamp clock source select
0 - time stamp clock source is on-board 1us clock
1 - time stamp clock source is event #6

--

7000,0007: rd only Config 1 register
bits 6..0 - analog multiplexer settling time
1 count = 4*pclk = 121.21 ns
programmable time = 0 to 7.636 us

4.12.3 Data FIFO

8000,0000 : rd Read event code from event FIFO
8000,0000 : wr Clear FIFO

4.12.4 Event FIFO

9000,0000 : rd Read event code from event FIFO
9000,0000 : wr Clear FIFO

4.13 On-Board 80960 CPU Hardware Interrupts

INT 0 Sync Event Interrupt
Data sync event from timeline.
The CPU will copy current scan count and pointer info for reading by VME, then the CPU will send sync interrupt to VME.

--

INT 1 1 ms Clock Interrupt
This interrupt is used by the CPU to time Arm trigger delays and Halt trigger delays. When this interrupt occurs, the CPU will also read the 'Data Fifo Empty Bit' to

determine if data is available in the FIFO.

--

INT 2 Data FIFO Half Full Interrupt

This interrupt notifies the CPU that data is available in the FIFO.

--

INT 3 Event FIFO Interrupt

This interrupt notifies the CPU that a timeline event is available in the FIFO.

--

INT 4 Arm/Halt Trigger 0 Interrupt

This interrupt notifies the CPU that the scan group 0 Arm/Halt trigger event has occurred .

--

INT 5 Arm/Halt Trigger 1 Interrupt

This interrupt notifies the CPU that the scan group 0 Arm/Halt trigger event has occurred .

--

INT 6 VME Command Interrupt

This interrupt notifies the CPU that the a VME command needs processing. A memory location in shared memory will be dedicated for this function. VME will first write a command request to this dedicated memory location, then issue a VME command Interrupt. The CPU will read and process the command, then write a return code to another dedicated memory location in shared memory. VME will either poll the return code memory location, or wait for the command complete interrupt to determine that the command processing has completed.

5.0 Software Control Descriptions

5.1 Sending Commands from VME to On-Board 80960 CPU

The front end computer sends commands to the MADC Controller module by first writing the appropriate command code and associated parameters to the command structure ’vme_command’ in shared memory. Then the front end compuer writes to register 0x2a00d to interrupt the on-board 80960 CPU. When the 80960 CPU program detects this interrupt, the command structure is read, and the requested command is executed. When command execution is complete, the 80960 CPU program writes a response code to the structure 'vme_response' in shared memory, then issues a VME interrupt to indicate that the command is complete. The front end computer reads the response code to determine if the command was executed successfully.

The front end computer may either poll the response code memory location, or wait for the command complete interrupt to be received. Before issuing each command, the front end computer should first initialize the response code to some unused value. This can be used to verify that the 80960 CPU has updated the response code.

Following is a list of valid commands. The include file madc2.h defines the command codes and response codes.

5.1.1 Start Acquisition

Command code: START_ACQ

Parameters: group number

When this command is recieved, the i960 CPU program reads the buffer setup structure for the requested scan group number, and then starts acquisition based on the parameters in the setup structure.

Valid values for group number: 0, 1

Valid response codes:
    CMD_DONE
    INVALID_CMD_CODE
    INVALID_GROUP_NUM
    INVALID_ARM_CODE
    INVALID_HALT_CODE
    INVALID_DATA_FORMAT
    INVALID_INT_MODE
    BUFF_TOP_PTR
    BUFF_SIZE
    NO_CODE

5.1.2 Stop Acquisition

Command code: STOP_ACQ

Parameters: group number

When this command is recieved, the 80960 CPU program stops acquisition for the requested scan group number.

Valid values for group number: 0, 1

Valid response codes:
- CMD_DONE
- INVALID_CMD_CODE

5.2 Data Synchronization Across Multiple Boards

The following scheme has been developed to provide data synchronization across multiple boards during continuous scanning. Refer to the figure.

The front end computer will be interrupted on every occurrence of a dedicated sync event (maybe once every 5 seconds). On this interrupt, the front end computer will read the memory locations containing 1) the number of scans at time of last sync event and 2) the data pointer at time of last sync event. The number of scans will be added to the ’total scan count’ in the front end computer. This ’total scan count’ represents the number of total scans since acquisition for the scan group was started. An independent count will be accumulated for each scan group on each controller module. This scheme allows data to be synchronized across all MADC controller boards in the system on every occurrence of the sync event. If one board is restarted, then all boards will be synchronized again on the next sync event.

Chapter 3

Site Wide Names

Although more than one name may apply to a RHIC object, the preferred name, especially in the context of controls software, is the **SiteWideName**. SiteWideName conventions are similar, but somewhat different, in the Blue and Yellow rings of RHIC.

3.1 Naming conventions

/RHIC/ATR/sitewidename.html

This is the "root" page that leads to a description of **SiteWideNames**, and naming conventions, in the ATR:

Site Wide Names

SiteWideName's for the ATR may be divided into several categories:

- Tunnel elements in the ATR.
  - * beamline optical elements (magnets, etc.)
    - Conventions for SiteWideName's of beam line elements.
    - g-2 Primary V-line elements.
  - *beam loss monitors
  - *beam dump
  - *vacuum components
- Other SiteWideName's.
  - *electronics racks
  - *power supplies
    - Conventions for SiteWideName's of magnet power supplies.
  - Timing system
    - *Delays
    - *Events (RHIC event link)
    - *Events (RTDL)

Note: Selections with a preceding "*" are database queries.

There is also a SiteWideName browser which queries the **atr_gddb** database.

Mangled by Waldo MacKay (waldo@bnl.gov).
Last update: 25 June, 1995

3.2 SWN conventions for power supplies

/RHIC/ATR/psswn.html

Site wide names of magnet power supplies are usually formed by adding a "ps" to the front of the magnet's SiteWideName except in the following cases:

- **psuarc4** for the 4 degree dipole bus in the U-line consisting of **ud1** and **ud2**.
- **psuarc8** for the 8 degree dipole bus in the U-line consisting of **ud3** through **ud6**.
- **pswarc20** for the 20 degree dipole bus in the W-line consisting of **wd1** through **wd8**.
- **psxarc90** for the large arc in the X-line with dipoles **xd1** through **xd31** and lambertson magnet **xlamb**.
- **psyarc90** for the large arc in the Y-line with dipoles **yd1** through **yd31** and lambertson magnet **ylamb**.
- **psxd31t** for the trim 100A bipolar supply across **xd31**.
- **psyd31t** for the trim 100A bipolar supply across **yd31**.
- **psxlamt** for the trim 100A bipolar supply across **xlamb**.
- **psylamt** for the trim 100A bipolar supply across **ylamb**.

All the rest of the ATR elements are powered by single supplies with a one-to-one correspondence.

3.3 SWN conventions for optics

/RHIC/ATR/swn − optics.html

SiteWideName's for beamline components

Magnets for the ATR fall into several categories: horizontal dipoles, vertical pitching magnets, quadrupoles, trim dipoles, and lambertsons. Other beam line elements include a stripping foil, flags, beam position monitors, current transformers, collimators, vacuum equipment and beam loss monitors. Within the AGS and RHIC rings are also ferrite kicker magnets.

There are several different types of names which are used for each element within the definition of the lattice, and for survey information, but there is a unique SiteWideName for each element which should be used at the top level of the control system and for operations.

SiteWideName's for elements in the transfer lines begin with the name of the beamline (u, w, x, or y) in which the element resides, with the exception of the switching magnet upstream of the beam dump. For logical reasons dealing with tracking programs, the switching magnet is considered to be in both the X- and Y-lines, so we have decided to give it the SiteWideName: "swm" for switch magnet. Note that SiteWideName's are typed in lower case on the computer.

The rest of the SiteWideName's for the transfer lines are generally made up of three or four fields:

```
        beamline dev_type number mod
```

where *beamline* is the single letter designator for the beamline, *dev_type* is a mnemonic for the type of element, and *number* is sequence number of the particular element within the specified beamline. The modifier *mod* is used at present only for collimator jaws and some vacuum components.

Since there is only one stripping foil, there is no *number* in its SiteWidename.

Device types for typical ATR elements are as follows:

```
  Magnets:
        d       horizontal dipole (with or without gradient)
        p       vertical pitching dipole (no gradient)
        q       quadrupole
        th      horizontal trim steering magnet
        tv      vertical trim steering magnet
        lamb    lambertson magnet
        swm     switching magnet (pure dipole)
```

```
Beam monitors:
      bh      single plane beam position monitor (horizontal)
      bv      single plane beam position monitor (vertical)
      b       dual plan beam position monitor (horizontal & vertical
      f       flag profile monitor
      xf      current transformer

Collimators and foils:
      foil    stripping foil
      c       collimator

Vacuum components:
      ip      ion pump
      rv      roughing valve
      sv      sector valve
      fv      fast valve
      cc      cold cathode gauge
      tc      thermocouple gauge
```

(A larger official list of RHIC mnemonics is available.)

The sequencing of trim magnets does not distinguish between horizontal and vertical, even though the *dev_type* does. The sequencing for beam position monitors is similar to trim magnets. Sequencing of vacuum components is done by setting *number* to the sector number within the beamline for the given element and using a letter *mod* beginning from A for the upstream end of the sector and running down the alphabet for identical devices within the sector.

The modifiers for collimator jaws are "l" for a left jaw and "r" for a right jaw.

Since the V-line for the g-2 experiment branches off from the U-line, just upstream of the 8 degree bend, we refer to the two V-line dipoles sharing a common vacuum chamber as VD3 and VD4.

Mangled by Waldo MacKay (waldo@bnl.gov).
Last update: 22 June, 1995

Chapter 4

Application Software

/RHIC/RAP/Documentation/Apps/root.html

This is the "root" page for application software documentation:

RHIC Application Software Documentation

RHIC Accelerator Physics (RAP)

Last modified September 21, 1995, by Satogata

	Section	**Description**	
1.	Introduction	Introduction to this documentation, brief explanations.	
2.	Projects	The list of current application projects, with documentation links.	
3.	Managers	The list of current managers, with documentation links.	
4.	Software release procedures	How to release applications, glish scripts and pet files	**New**
5.	Applications review notes	Notes from ongoing applications reviews, including style comments and comments on individual applications.	
6.	APPS styles and rules	Including ProjTemplate, file structure, and brief notes on the console environment.	
7.	Environment variables	Relevant ENVIRONMENT variables, and current assignments.	

Maintained by Todd Satogata / (516) 282-5452 / satogata@bnl.gov
Last Update: **September 21, 1995**

4.1 Introduction

/RHIC/RAP/Documentation/Apps/introduction.html

RHIC Application Software Documentation: Introduction

RHIC Accelerator Physics (RAP)

Last modified September 11, 1995, by Satogata

Location

The software development area for RHIC (and ATR) applications and managers is on the RHIC Accelerator Physics (RAP) fileserver, "owl.rhic.bnl.gov". It can be reached with the command "`cd $APPS`" on development workstations. *$APPS* is a UNIX environment variable set at login on the "zoo" workstation domain.

Directory Structure and Style

Each application developer is assigned a **Project** directory inside which one or more applications are developed. The structure follows that in the 'project template' directory *$APPS/ProjTemplate* - each top-level project directory has subdirectories which contain source code for applications or libraries, public and private header files, application-specific startup data and Xt app-default files, etc. (Contact Todd Satogata for more info.)

Similarly, manager developers develop in directories below the *$APPS/manager* directory, and libraries which are used by a variety of applications developers may be placed in the directory *$APPS/libraries* to be shared.

There is a brief style guide that should be followed by applications developers. This style guide includes such conventions as where both generic and application-specific data files are located with respect to environmental variables such as *$RELEASE_DIR*, as well as comments on installation and release procedures. This style guide is, by necessity, a live document, and constantly evolving.

Prerelease, Release, Replication

At appropriate intervals the **appmeister** (Todd Satogata, satogata@bnl.gov), performs an

apps release of all the applications and managers. This places the executables, libraries, et cetera, into the area /ride on the RHIC Controls Section development file server, cfsa.rhic.bnl.gov, along with myriad software developed and maintained by the RHIC Controls Group. The **appmeister** is also responsible for all other technical aspects of the $APPS area.

Whenever necessary, the current **release** of controls software is **replicated** onto the disks of the operational consoles in the */usr/controls* directory. Scripts used to perform the replication are the responsibility of Rich Casella (rac@bnl.gov) - they copy and **strip** application binaries, scripts, app-specific data and app-default files, as well as canonical data. *(Question: what about documentation?)* They explicitly do not replicate libraries or source code.

Part of the application look-and-feel is determined by the default console environment characteristics set by the **mcr** user, since all consoles are logged in as him/her/it. The default **mcr** environment is the joint responsibility of the zoo system administrators, Rich Casella and Roger Katz (roger@bnl.gov).

Maintained by Todd Satogata / (516) 282-5452 / satogata@bnl.gov
LastUpdate: **September 11, 1995**

4.2 Projects

/RHIC/RAP/Documentation/Apps/projects.html

RHIC Application Software Documentation: Projects

RHIC Accelerator Physics (RAP)

Last modified September 20, 1995, by Satogata

The following is meant to explicitly follow the organization of **pagetree** on operations consoles during the 1995 ATR commissioning run. If this is not the case, please email or call Todd Satogata.

Links in the **Application** column point to brief descriptions of each application and project following this list. Links beneath the **Doc Update** column point to online documentation provided by the applications programmers and systems commissioners.

	Application	**Type**	**Developer**	**Phone**	**Doc Update**	**App Release**
0.0	pagetree	GUI	Waldo MacKay	3076	7 Sep 95	?
1.0	plot_atr	GUI	Waldo MacKay	3076	7 Sep 95	?
2.0	Magnet Control/Display	GUI	Jorg Kewisch	5653	21 Sep 95	?
2.1	PS Digital Control	PET	Jorg Kewisch	5653	21 Sep 95	?
3.0	Orbit Display	GUI	Todd Satogata	5452	**Never**	?
4.0	Event Link	PET	Waldo Mackay	3076	**Never**	?
5.0	BLM Display	GUI	Steve Tepikian	4845	**Never**	?
5.1	BLM Digital Control	PET	Steve Tepikian	4845	**Never**	?
6.0	MADC Control	PET	Jorg Kewisch	5653	**Never**	?
7.0	Current Display	PET	Dong-Ping Deng	2197	19 Sep 95	?
8.0	Profile Monitor Display	GUI	Ping Zhou	7779	15 Sep 95	?
Utilities						
1.	Pet Program	GUI	Don Shea	2931	Feb 28 95	?
2.	tabdis	GUI	Waldo Mackay	3076	7 Sep 95	?

Descriptions of applications and projects

pagetree
Graphical startup of RHIC/ATR applications programs.

plot_atr
Display of ATR beamline, including site-wide names, magnet outlines, tunnel walls (where available) and site coordinates.

Magnet Control and Display
Graphical interface for tweaking and setting ATR magnet strengths and currents. Step-stone storage and retrieval allow primitive user-driven ramps to be constructed.

Power Supply Digital Control
Control of status, on/off settings and other digital power supply readbacks from a pet page, or set of pet pages.

Orbit Display and Archival
Display and archival of beam trajectories along ATR, including position information from flags when applicable.

Event Link Pet Page
Pet page displaying event codes for the ATR REL (RHIC Event Line), as well as channel control and status for event encoders.

BLM Display and Control
Display and archival of beam loss monitor readings along ATR.

BLM Digital Control Pet Page
Control of BLM integrator timing, power supply settings, readback and status.

MADC Control/Graphing Pet Page
MADC Control/Graphing Pet Page

MADC Control/Graphing Pet Page
MADC Control/Graphing Pet Page

Transformer Control / Current Measurement
Transformer Control / Current Measurement

Profile Monitor Display
Profile Monitor Display

Pet Program
Pet Program

tabdis -- glish tabular display
tabdis -- glish tabular display

Maintained by Todd Satogata / (516) 282-5452 / satogata@bnl.gov
Last Update: **September 20, 1995**

4.3 Managers

/RHIC/RAP/Documentation/Apps/managers.html

RHIC Application Software Documentation: Managers

RHIC Accelerator Physics (RAP)

Last modified September 11, 1995, by Satogata

This page lists the current **managers** in development by, or available to, applications programmers for ATR and RHIC. Managers are high-level controls programs which convert between relatively low-level FEC access (usually using ADOIF calls) and high-level physics-oriented descriptions of the accelerator suitable for applications programs (usually in glish).

The following is meant to be a list of managers in development for the 1995 ATR commissioning run. Please report any discrepancies by email or call to Todd Satogata, x5452.

Links in the **Manager** column point to brief descriptions of each manager following this list. Links beneath the **Doc Update** column point to online documentation provided by the manager programmers and systems commissioners, when available.

Manager	**Developer**	**Phone**	**Doc Update**	**Manager Release**
BPMs / Orbit	Todd Satogata	5452	**Never**	?
Loss Monitors	Steve Tepikian	4845	**Never**	?
Magnets / Power Supplies	Jorg Kewisch	5653	**Never**	?
Profile Monitors	Ping Zhou	7779	**Never**	?
Timing System	Unknown	????	**Never**	?

Descriptions of managers

Bpm/Orbit Manager
Provides hooks into sets of bpm control by site-wide names of bpms; includes control of gain settings, status readback and relevant timing control. Does not include modeling or hooks into profile monitor system.

Beam Loss Monitor Manager

Magnet Manager

Video Profile Monitor Manager

Maintained by Todd Satogata / (516) 282-5452 / satogata@bnl.gov

Last Update: **September 11, 1995**

4.4 Application release procedure

/RHIC/RAP/Documentation/Apps/release.html

RHIC Application Software Documentation: Release Procedure

RHIC Accelerator Physics (RAP)

Last modified October 4, 1995 by Satogata

Applications Software Releases

NOTICE

A scheduled replication onto *all* consoles is now being performed at 12:30 PM every Tuesday from the standard release area, `/ride/release/$(ARCH)` (`/usr/controls` on development consoles). Applications should be released whenever code changes result in new features being added, fixed or changed -- they should not be driven by a schedule of console replications. Intermediate replications will be performed at the discretion of the replicators.

Release Procedure for Applications

The following procedure must be followed by applications developers to release their softare into `/usr/controls` on development consoles when necessary. Please call Todd (x5452) or Smita (x3924) if you have any questions or concerns, or to get the user password for the 'release' account. This is a modified version of the release procedure documented by the RHIC Controls group in the file "/ride/documentation/releaseProc.fm.html".

1. As yourself, build your application normally by typing 'make' on a development SUN architecture machine (or whichever architecture you happen to want to release for.) This forces a cleaner compilation, as well as avoids file protection problems and conflicts with the 'release' user later. Test your application locally as you deem necessary.

2. Change any menuTree paths in your app-defaults files to point down data directories in /usr/controls, the final replication path. There are cleaner alternatives to this -- the release makefiles automatically substitute environment variables of the

form `${ENV}`; if you use these files, talk to Don or Todd.

3. Switch users to the 'release' user with the command "`su - release`" The '-' reads the entire environment such as .cshrc and .login. See above for who to contact for this password.

4. Type the command "`openPerms`". This opens permissions on the /ride/release directory to the 'release' user.

5. Check that your `RELEASE_DIR` is `/usr/controls`. If you are feeling particularly fretful, check that this is a symbolic link to `/ride/release/$(ARCH)`.

6. Type the command

   ```
   make release
   ```

   to release the program.

7. Type the command

   ```
   closePerms
   ```

   to close the permissions to the release directory.

8. Email relevant users and <satogata@bnl.gov> to inform people of the release.

Pet Page and Glish Script Release

There are still unfortunately a couple of loose ends regarding release and replication of PET pages and glish scripts. My suggestion is to keep these scripts and files in the 'data' area just like other data for applications. For an example of the (2-line!) Makefile to release pet pages in this way, have a look at the directory

```
/apps/orbit/BpmPet
```

The Makefile here -- which can be copied *without change* for other pet areas -- will release pet pages into `${RELEASE_DIR}/data/PetPages`. This can be edited to also release into subdirectories, so in the future we can organize Pet page releases and replications as we see fit.

Glish scripts should be kept in the 'data' area of the application that they were written for; e.g. `OrbitDisplay.g` is application-specific, and is released to the data/OrbitDisplay area as the tree file for this application is.

Testing Released Software

Once released, controls and applications software may be tested a variety of ways without the requirement of replication to a console. The `atr` and `mcr` users each use released software immediately on development consoles (as opposed to operations consoles, which require a replication). Therefore, a reasonable procedure to test newly released software is to run `pagetree` on a development console as one of these users, and see if your software runs, including setup files, pagetree data file entries, etc. If the pagetree data file startup needs to be changed, contact Todd.

Prerelease testing may also be done at a similar level by checking that your `RELEASE_DIR` environment variable is set to `/ride/prerelease/$ARCH` on a development console, then testing as usual.

I would greatly appreciate any suggestions regarding more robust or appropriate software testing procedures.

Comments?

Maintained by Todd Satogata / (516) 282-5452 / satogata@bnl.gov
Last Update: **4 Oct 1995**

4.5 Software review notes

/RHIC/RAP/Documentation/Apps/review.html

RHIC Application Software Documentation: Review

RHIC Accelerator Physics (RAP)

Last modified September 11, 1995 by Satogata

Application Environment Review Comments and Updates

The following is a list of suggestions and comments about applications programs standards and development from ATR crew chiefs (Steve Peggs, Waldo MacKay, Dejan Trbojevic, Jie Wei, Jorg Kewisch), AGS operators (Jon Laster, Willem VanAsselt) and Todd Satogata on Friday, August 25 1995. They have been updated to include comments from another later meeting with Leif Ahrens and Ken Reece of the AGS. Starred items (*) are considered critical; (?) items are considered quite low priority.

These pages will be updated routinely to include continued commentary on applications as well as updates from developers when issues are addressed. Application hypermail resources, when available, will supplant this page in the future.

	Section Name	**Developer**	**Phone**	**Web Doc Updated?**
1.	General Comments for All Apps	All Developers		**11 Sep 95**
2.	Pet Pages	All Developers		**20 Sep 95 - some**
3.	Pet Program	Don Shea	2931	**Never**
4.	Page Tree	Waldo MacKay	3076	**7 Sep 95**
5.	Plot_ATR	Waldo MacKay	3076	**7 Sep 95**
6.	MagControl and MagDispl	Jorg Kewisch	5653	**Text**
7.	Orbit Display	Todd Satogata	5452	**Never**
8.	BLM Display	Steve Tepikian	4845	**Never**
9.	Profile Monitor App	Ping Zhou	7779	15 Sep 95
10.	Current Monitor App	Dong-Ping Deng	2197	19 Sep 95
11.	Event Link Pet Page	Waldo Mackay	3076	**Never**
11.	MADC Control Pet Page	Jorg Kewisch	5653	**Never**

General Comments -- All Apps

- ***** Help Menus and Documentation, Third-Button Help**
  Help menus and online documentation *must* be improved; third-button help popups is a convention that both AGS and pet pages are already using. It is possible to put attributes into pet startup files for internal documentation of pet pages (See Don Shea for help on this), as well as to put online help into the app-defaults X11 resource file of applications. (See Ted D'Ottavio for help on this.) Help on specific devices from within applications would also be, well, helpful (from Jon Laster.)

- ***** Information Bar: Displaying Date, Time and Cycle Info**
  Where possible, dates and times should be displayed on pages. Jon Laster suggested that we design an 'information bar' that could be incorporated into applications underneath their menu bars, including info such as date, time, AGS cycle number, status (including status of whatever current dataset is being displayed) and perhaps application-dependent status information and an editable comment.

- ***** Consistent Labeling of Fields and Graph Axes**
  Where units are applicable, *use them*! Also use graph legends to label graph data, and sidebar windows if displays are too cluttered.

- ***** Dataset Save Conventions**
  All saved datasets should include a *comment* field, browsable by a suitable application at a later date. This may develop to a file browser that displays comments and file types along with filename info, but this is a long-term project. For now, popups that request and display this comment field upon save or load are enough. Maybe. Included in the save should also be relevant information from the information bar, above.

- ***** Consistent Use of Environment Variables**
  Environment variables should not be assumed to be present -- i.e. the `getenv()` return should be checked to be *sure* it is non-NULL. (Using this without checking often results in hard-to-debug segmentation faults when an environment variable is not set, as Todd knows from experience!) Environment variable standards should be used for startup, canonical, volatile, application storage and log data.

- ***** Unselecting Pull-Down Menus**
  Currently pull-down menus evidently cannot be unselected once selected, save for selecting one of their entries. This is a topic of concern for Ted D'Ottavio, as it's a UI Tool problem.

- ***** ADOs Should Be Checked for Async Updates**
  Some ADOs currently in the field update information asynchronously when they are changed -- a good practice, as this updates pet and any other application with an async registered on that ADO. We should ferret out the ones that don't comform to this and change them.

- **Use a Message Area for Messages**
  If you have a message area in an application, *use it* for informational and warning messages. Additional methods may be added to the UIMessageArea object to handle color changes for warning situations.

- **Use `rhicErrors` for ADO/Other Error Reporting**
  Talk to Don Shea about this if you're not sure what it is; rhicErrors is a class library that gives users access to string representations of common ADO error codes. (Don: Could this be incorporated into the libadoIf?)

- **Provide Tabular Formats For Data Display**
  Jon Laster and other operations folks would like tabular displays of array data as well as graphs for BPM and BLM information, as well as any other info that presents itself in this manner. The 'pet' method of tabular display can likely be used with very few changes.

- **Conform for Any Means of Application Exit**
  Traditionally, Ted has build applications using TreeBuilder and the UI Toolkit that `exit()` immediately if the 'Close' button is selected from the upper left window manager pulldown menu. (In UI programs, this is capturing a UIWindowMenuClose event.) All methods of exiting an application (with the possible exception of glish 'terminate' and NULL events) should pop a confirmation window. Also, from Jon Laster, this confirmation window should allow a 'Return' keypress to confirm exiting.

- **Keyboard Accelerator Standards**
  Common conventions should be established for standard control keys within applications. (e.g. Ctrl+Q is 'Quit...', Ctrl+S is 'Save...', etc.)

- **One-Field Two-Click Confirmation Scheme**
  The confirmation method for pet (click on a field with left button to hilite, then click again with middle button to send or confirm) is considered to be a good one, and should be followed wherever possible when fields or buttons send settings to hardware.

- **Glish Event Conventions for User Interfaces**
  Tying user interfaces together later, if necessary, will be much easier if we put the hooks in with the first generation. This implies a standard set of UI and glish events for such things as window resizing, graph rescaling, etc. Steve Tepikian, Don Shea and Todd are currently discussing addressing this issue for the BPM and BLM display programs.

- **Zooming Clicks on UIGraph**
  By default zooming should be turned *off* on UiGraph objects, only to be turned on explicitly with the right mouse button menu. (A help menu could also be added underneath that menu, btw.) Since the control environment is click-focus, a focusing click on an application's graph window can also accidentally zoom the graph.

- **??? Expert, Help and Print Buttons on Menu Line**
  Some technical options (such as ADO communication and information) can be

relegated to an 'expert' menu, placed at the upper right of application menu bars if present. Standard utilities that every application supports, such as printing and help, may also be placed here, though for '95 ATR tests third-button help and PrintTool printing will probably have to suffice.

Pet Pages

- ***** Some Critical Cells Should Default to Non-Editable**
  E.g. cells such as the timeline event codings on the events page. It's easy enough to set a cell editable again if really necessary, but cells where one entry has system-wide impact or is very sensitive should definitely be made to default to non-editable. This can be set up in the pet page startup files on a page-by-page basis.

- **??? Pet Page Documentation**
  According to Don, there is currently no mechanism for user-supplied help in pet (i.e., specific to a particular pet startup page.) Doc for pet pages should be arranged on the web for '95, with the addition of a 'help' button popping a user-supplied help file later.

Pet Program

- ***** FEC Down Causes Pet to Hang**
  When pet fires up and attempts to contact a crate which is offline (either turned off or encountering network problems), it hangs with no status information displayed. A timeout might be included or some other alarm to inform the operator that the particular front end pet is failing to contact is down.

- ***** Pet Does Not Warn on FEC Name Resolution Failure**
  When pet starts on some consoles which have NIS problems, and does not resolve the fec name properly to an IP address, it displays the pet page with completely blank fields for data. This is an error state and pet should warn the user, telling which FEC name it is having trouble resolving (if possible).

- **Deiconify Event Should Trigger 'Get All'**
  When deiconifying a pet page, the user likely wants 'live' data. Catching the

UIDeiconify event and throwing a 'get all' to refresh the data on the page is a good idea -- alternatively a standard operations procedure could be to type ^A when deiconifying any iconfied pet page.

- **Unsigned Byte Display Problem**
  Pet has a bug on Suns that displays unsigned bytes as though they were signed, possibly causing confusion when displaying REL event codes and other unsigned byte data. This is actually related to a Glish/Value class problem, it seems.

- **Field Selection and Editing Inconsistencies**
  Users can select cells that have no data in them, and 'edit' data in so-called uneditable cells (that is, point, click and type -- pet won't let you actually *change* the value, however). These are minor but valid annoyances.

Page Tree

- ***** Double Clicks to Start Applications**
  Using single clicks to start applications violates established conventions and is generally annoying since user often start two applications with a double-click. Use double-clicks on an app name to start up a given application.

- **`stderr/stdout` Are Used For Error Reporting and Not Directed**
  Either an xterm could be popped with pagetree to report statuses and errors, or -- more appealingly -- a scrolling text area could be added to pagetree, much in the same manner as 'startup'. This way application startup information and errors are all shown in one log.

- **??? Change Application Start Messages**
  Currently pagetree prints a message in the message field that is the name of the application just started. Include something in this message such as 'Starting' to let the user know it's actually starting the bugger, and not just an informational message.

- **??? Name Fields Do Not Resize With Window Resize**
  This is really picky, actually, but true; could you possibly catch a stdwindow UIResize event?

Plot_ATR

- ***** Change Verbosity**
  Allow a command-line switch to turn on (or off) the coordinate printout when zooming, or keybind a key for this info. This info stands a good chance of clogging the pagetree log.

- **Display Coordinate Ranges Onscreen**
  Perhaps a small label showing currently zoomed coordinate ranges could be done easily, in one of the upper corners. This actually goes along with the previous comment, and might be a better implementation.

- **Supply Online Help; `stderr/stdout` Are Currently Used**
  It would be nice to have a display of online help somewhere instead of having to find the controlling xterm. Suggestions include a small modal window popup (from pressing 'h') or another app, 'plot_atr help', that's started at the same time plot_atr is.

- **Set Limits for Zooming**
  Zooming too far in one direction or the other creates something ugly. Best just to check limits and lock out at some point, either when too small (scale on order of meters) or too large (scale on order of tens of thousands of meters.)

- **??? Glish Interface**
  It would be *very* keen to have a plot_atr interface where someone passes a SWN in and it zooms to that spot in the beamline. Probably too much to ask at this point, though, since there's currently no provision for a combined X event/Glish interface without UITools.

MagControl and MagDispl

- ***** DOIT With Middle Mouse Button Crashes App**
  This appears when this button is first hilited with the left mouse button, probably 'armed'. Error returned is an X 'BadValue' error.

- ***** Change Units In Some Places, Foil Strategy**
  'Energy' should be 'Gamma', for example. Is this energy/gamma before or after the stripping foil, and could some control of the stripping foil strategy be

included?

- ***** Dependent/Independent Variables Are Not Clear**
  When one scales energy, one usually means to keep the kick strength *constant* but vary the the underlying current to the magnet. This is not what is done currently -- energy (gamma) scales even the kick strength in mrad. This is likely not correct for procedures like measuring dispersion.

- **Display Comment or Label for each Step-Stone**
  If for no other reason than to clarify matters, especially when printing.

- **Display Power Supplies and Magnets in Beamline Order**
  That is, display them in their order down the beamline instead of alphabetical order.

- **Loading 'Live' Field Readings?**
  Can one load the current settings from the field into magdisp? This would be very nice to do for small tweaks and changes to the machine unless 'doit' button only sends fields that have changed. If this is the case, hilite the fields that have changed.

- **Save and Load to *APP_STORE/APP_VOLATILE*; Saving Ramp Groups?**
  Currently this application saves and loads to/from Jorg's home area. Ramps (sets of step-stones) should be set up in the application *$APP_STORE* area by Waldo and Jorg for hysteresis cycling. It would also be very nice to be able to store sets of step-stones together, but this can be faked using directory structures for '95.

- **Asynchronous Status Readbacks?**
  Is the status readback asynchronous on this page, or just the PS digital control pet page?

- **??? Combine Two Displays Into One**
  It would be nice, although there is some shared data between the two displays and some disparate data (such as file load and save).

- **??? Field Style Comments and Changes**
  Default fields could be widened to include typical information such as error code readbacks. Also, do you really like blue and black text?? :-)

- **??? Include Power Supply Label Fields on Magnets to Show Busing?**

A pie-in-the-sky thing, really, but it would be nice to see a list of which magnets are on which power supply somehow. This would certainly help debug wireup problems that are database-driven.

Orbit Display

- ***** Display Plane Information, One or Both Planes**
  There is no label telling the user which plane(s) they're working with. A second graph should be added when doing dual-plane work as well.

- ***** Display BPM/Flag Locations**
  BPM and flag locations and SWN's could be included on the beamline graph. Flags should also be displayed in a different color if possible, possibly with 'in' or 'out' status info.

- ***** Display Mode Information and BPM Gain Settings/Statuses**
  BPM gains and statuses should be displayed in a table somewhere on the application screen. This table may even be editable (probably not though), but it should at least display, one one screen, all BPM gains and statuses together for printing and perusal. The acquisition mode and operational plane (pretty apparent with the above suggestion) should be displayed as well.

- **??? Include Running Differences**
  Include a shot-to-shot difference mode? This is posibly useful for diagnosis.

- **??? Display Line Labels on Beamline Graph**
  It would be nice to have labels for the various lines (U, W, Y) on the beamline graph (Jon Laster).

BLM Display

- ***** Display Mode Information, BLM Gain Settings/Statuses**
  BLM gains and statuses as well as power supply statuses and settings should be displayed in a table somewhere on the application screen. This table may even be editable (probably not though), but it should at least display, one one screen, all current BLM gains and statuses together for printing and perusal. The

acquisition mode and should be displayed as well.

- ***** Labels Need More Consistency and Coherence**
  Vertical scale units are probably ’counts’. Multiple plots are not labeled, nor are any plots labeled with the buffer number or any sort of identifier to discriminate them.

- ***** Clarify Menus, Include Mode Settings Under ’Acquisition’ Menu**
  What are ’Start’ and ’Stop’ buttons? What are the buffer buttons good for? Why should you ever empty the buffer -- is it rotating? Have a look at Orbit program with continuous and single-shot acquisition modes.

- ***** What Data Is Saved? Gain Settings and Power Supplies?**
  Gain settings, power supply settings and comments should be stored with every data set.

- **Differences from Reference and Running Differences**
  As in the BPM/Orbit program, running differences and differences from a given set of reference readings would be very helpful.

- **??? Default to Logarithmic Y Axis?**
  This may or may not be a better default than the current linear axis; operational experience will decide this for us.

Profile Monitor App

- ***** Review comments forthcoming**

Current Monitor App

- ***** Review comments forthcoming**

Event Link Pet Page

- ***** Review comments forthcoming**

MADC Pet Page

- ***** Review comments forthcoming**

Maintained by Todd Satogata / (516) 282-5452 / satogata@bnl.gov
Last Update: **11 Sept 1995**

4.6 Styles and rules

/RHIC/RAP/Documentation/Apps/styles.html

RHIC Application Software Documentation: Style Guide

RHIC Accelerator Physics (RAP)

Last modified September 11, 1995, by Satogata

The various styles and rules associated with applications development are intended to help, rather than hinder, the developer. Unnecessary complication has been avoided and the consensus of the Continuing Task Force has been followed, wherever possible. Of course, the developer is free to ignore some of the apps conventions (at his own peril) so long as the true goal of guaranteed performance in the control room is met. Control room performance criteria include, for example, *reliable and consistent data storage* and *consistent look-and-feel*.

Makefiles and $APPS/ProjTemplate

Included in the $APPS area is the template directory $APPS/ProjTemplate. This area typifies the recommended organization a simple application's development area. Beneath each top-level application directory (e.g. *$APPS/Orbit*), there are subdirectories for each library or set of associated binaries. Sample application library and binary makefiles (quite simple, in fact) are available underneath the $APPS/ProjTemplate area. There are also areas that are shared between several different applications, including

$APPS/libraries
Libraries shared among applications, including those for string manipulation (mackay@bnl.gov), lattice display (tepikian@bnl.gov) and dynamic glish connections (jorg@bnl.gov).

$APPS/manager
Glish-based high-level managers for subsystems, such as BPMs, magnets and BLMs. Manager documentation is forthcoming; systems commissioners and application programmers have more information.

File Structures and Data Management

The application file structure distinguishes between **volatile**, **archival**, **start-up**, and **canonical** data. Correct use of the application file structure is made easier by the availability of various environment variables. The default console environment also constrains the developer -- for example, **volatile** files saved to console disks are not backed up, and are not readily available at other consoles.

Database Access

Console Environment

Following the tradition of the Apollo computers used for AGS control, all consoles used for RHIC and ATR operations will be logged in as the user **mcr**. The console environment determined in large part by the **mcr** user is different in various ways from the AGS/Apollo paradigm, both in terms of the standard set of windows (or icons) presented, and in terms of philosophy of use. For example, in 1995 it will be possible to launch ATR applications and pet pages from a default pagetree menu. If the **startup** utility familiar to Apollo users is ported to the Suns, it too will be available for launching. The document "AGS and RHIC Environment" by Tom Clifford summarizes several other environmental f eatures - such as mounts, dependencies, and database servers - that have been aired in Continuing Task Force meetings.

Maintained by Todd Satogata / (516) 282-5452 / satogata@bnl.gov
LastUpdate: **September 11, 1995**

4.7 Environment variables

/RHIC/RAP/Documentation/Apps/env.html

RHIC Application Software Documentation: Environment Variables

RHIC Accelerator Physics (RAP)

Last modified September 11, 1995, by Satogata

Application Data Storage Conventions

There are at least five different classifications of data used by applications in the APPS area -- startup, canonical, volatile, application storage and log. These data areas should be referenced with environment variables (using the `getenv()` system call) and application-names (#def'd in header files if possible) within high-level controls applications, as specified in the following list. Directory organization beneath these areas is application-specific.

- **Startup data**: $RELEASE_DIR/data/<app-name>/
  This is where tree files and application-specific startup data files are located upon release from the $APPS area. These data files are released and replicated to control consoles at the same time and in the same manner as applications programs. (e.g. `pagetree/rhic.pages`.)

- **Canonical data**: $CANON/<database-name>/
  This is where database tables and groups common to many applications programs are located upon release from the $APPS area. Currently this area contains **holy_lattice** and **atr_gddb** directories, including design optics, aperture and design information for the ATR. These tables are released and replicated to control consoles as well. (e.g. `holy_lattice/YTransfer/Namespace`.)

- **Volatile data**: $APP_VOLATILE/<app-name>/
  This is where 'live' and temporary data are stored while running a particular application. *This area is local to each console,* implying that volatile data is not accessible from other consoles, nor is it ever backed up. Examples are myriad, including error logs and reports, temporary simulation results and datasets, mirrors of SDS shared memory partitions, etc.

- **Application save data**: $APP_STORE/<app-name>/
  Application save data is data that is explicitly saved by the user or the application for backup and archival. All of this data is resident on **cfsb** (the control room file server), and exported and available to each console and applications thereon.

- **Log data**: $LOGS/<app-name>/
  Here log data means debugging and informational messages generated by applications. (This is in contrast to *logging* machine data, which is volatile or applications storage.) This data is stored locally on each console to avoid multiple-instance log interweaving. This is explicitly separate from volatile data for clarity.

APPS Developer Environment Variables

The following variables are used by programmers during development, especially within the application Makefiles:

- **ARCH:** Machine architecture, usually *SUN* or *SGI*.
- **APPS:** Top-level of applications area, usually */apps*.
- **RELEASE_DIR:** Release or shadowed testing directory, usually */usr/controls/* on control room consoles or */ride/release/$(ARCH)/* on development consoles.

See application conventions for more info on use of the following environment variables.

- **CANON:** Canonical data.
- **APP_VOLATILE:** Volatile (temporary and console-local) data storage.
- **APP_STORE:** Centralized backed-up data storage.
- **LOGS:** Log data, including informational and debugging messages.

Maintained by Todd Satogata / (516) 282-5452 / satogata@bnl.gov
Last Update: **September 11, 1995**

Chapter 5

Databases

The two main SYBASE databases used for ATR work are **atr_gddb** and **atr_cal**. This chapter describes the structure of their contents.

5.1 Generic Definition DataBase, atr_gddb

/www.rhichome.bnl.gov/cgi − bin/atr_describe.sh

Document generation time: Oct 11 1995 4:08PM

Description of the "atr_gddb" database.

This document shows what is currently in the database. (Some of the text descriptions of things may be old, if the the appropriate tables have not been updated.)

Quick index:

- Groups of tables and views
- Ungrouped tables and views
- Stored procedures

Groups of tables and views:

```
There are several useful tools for dealing with groups of tables.
(See the local UNIX man pages for "dbtools".)

Note to programmers:
Header files (for C and C++) can be found along the "$HORST/include" p
for most of these groups.  (The "header_info.h" file is located along
"$DBAPPLICATIONS/include" path.)  By including these paths in your
compilations, you should get the file (if it has been generated by "db
Functions for reading and writing SDS versions of the groups are locat
in "$HORST/lib/$ARCH/libgddb.a".
```

- beam_init
- blinit
- blm
- blout
- db_NameLookup
- fidgen
- generic_dev
- header_info
- html
- installation
- madc_channel

- magfield
- magnet_info
- picture_info
- ps_basic
- ps_ref
- ps_view
- real_dev
- SWNdesc
- table_desc
- tunnel_lines

Ungrouped tables and views:

- blm_serial
- BodyHarm
- bpm_orientation
- generic_bpm_info
- halfcores
- Integral
- MAD_type
- magnet_field
- magnet_field_bak
- movableBLM
- names
- nominal_bends
- ps_estimate
- ps_installation
- quad_transf
- SWNameAdo
- SWNameAdo_bak
- tcelement
- tunnel_arcs_bak
- tunnel_lines_bak

Stored procedures:

- dbg2sds_all
- describe
- fill_blm_avg
- html_describe_doc

- html_get_doclines
- html_group_list
- html_gt_list
- html_list_anchor
- html_p_list
- html_procedure_list
- html_t_list
- html_table_describe
- html_ungrouped_list
- initserial
- installed
- set_ps_limits
- undocumented
- what
- whatsit

Group *beam_init*

The C header file is "gddb/beam_init.h".

nominal beam parameters for various ion species (to be used with 'bl').

The tables and views in this group are:

- beam_init

Group *blinit*

The C header file is "gddb/blinit.h".

Lattice models and schemes for the "bl" program.

The tables and views in this group are:

- bl_lattices
- bl_monitors
- bl_tweaks

Group *blm*

The C header file is "gddb/blm.h".

BLM information about wiring and calibration.

The tables and views in this group are:

- blm_avg
- blm_HV
- blm_wire

Group *blout*

The C header file is "gddb/blout.h".

Definition of SDS objects for output from the bl program. These tables are just for the definition of the SDS structures and should not be filled in with values.

The tables and views in this group are:

- bl_elts
- bl_football
- bl_model_info
- bl_model_names
- bl_orbit
- bl_twiss

Group *db_NameLookup*

The C header file is "gddb/db_NameLookup.h".

This is an SDS header file for the NameLookup table in the atr_gddb database. The lengths of character strings are different from the definitions of Namespace made in the include files in $HORST/slotspace/include.

The tables and views in this group are:

- NameLookup

Group *fidgen*

The C header file is "gddb/fidgen.h".

Magnet fiducial information for installation.

The tables and views in this group are:

- fid_offsets
- magnet_survey

Group *generic_dev*

The C header file is "gddb/generic_dev.h".

Info for generic device descriptions and wireup.

The tables and views in this group are:

- Connectable
- Connects
- ConnectType
- Contains
- Device
- DeviceType
- Spigot
- SpigotOwn

Group *header_info*

The C header file is "Dbapps/header_info.h".

This lists how tables are collected into associated groups in SDS and *.h files.

The tables and views in this group are:

- header_groups
- header_tables

Group *html*

The C header file is "gddb/html.h".

This group contains some information for generating HTML documentation from the database.

The tables and views in this group are:

- htmldoclines

Group *installation*

The C header file is "gddb/installation.h".

Information for installation of magnets etc.

The tables and views in this group are:

- installation

Group *madc_channel*

The C header file is "gddb/madc_channel.h".

descibes which signal goes into which madc channel

The tables and views in this group are:

- madc_channel

Group *magfield*

The C header file is "gddb/magfield.h".

Shortened views to get magnet field vs current info.

The tables and views in this group are:

- magfield
- magquick
- ufoil

Group *magnet_info*

The C header file is "gddb/magnet_info.h".

Magnet information for installation.

The tables and views in this group are:

- magnet_data
- magnet_slot

Group *picture_info*

The C header file is "gddb/picture_info.h".

Information for drawing pictures of the beamlines.

The tables and views in this group are:

- generic_size
- otherelement
- pi_NameL

Group *ps_basic*

The C header file is "gddb/ps_basic.h".

Fundamental power supply definitions.

The tables and views in this group are:

- ps_data
- ps_limits
- ps_slot

Group *ps_ref*

The C header file is "gddb/ps_ref.h".

reference values for power supply information.

The tables and views in this group are:

- ps_ref

Group *ps_view*

The C header file is "gddb/ps_view.h".

A view to the power supply information.

The tables and views in this group are:

- ps_view

Group *real_dev*

The C header file is "gddb/real_dev.h".

Info for specific instances of devices related to the generic device descriptions and wireup.

The tables and views in this group are:

- PS_Mag_Wireup
- RealContains
- RealDevices

Group *SWNdesc*

The C header file is "gddb/SWNdesc.h".

Description of the object of a SiteWideName

The tables and views in this group are:

- SWNdesc
- SWNtable

Group *table_desc*

The C header file is "gddb/table_desc.h".

A table for describing database tables.

The tables and views in this group are:

- table_desc

Group *tunnel_lines*

The C header file is "gddb/tunnel_lines.h".

tunnel walls, etc., for plot_atr

The tables and views in this group are:

- tunnel_arcs
- tunnel_lines

Table: *beam_init*

beam_init is not listed in the "table_desc" table.

- examine

Table: *bl_elts*

The table "bl_elts" is just a template for an SDS object for the output from the "bl" program. The column names are:

```
        name:           SiteWideName of element,
        s:              s coordinate of element's downstream end,
        s0:             s coordinate of element'ss upstream end,
        mi_ind:         pointer into bl_model_info SDS object,
        e_ind:          pointer into element object of lattice file,
        l_ind:          pointer into lattice object of lattice file,
        type:           type of element from $LAMBDA/include/lattice_d
        NLtype:         type of element from $HORST/include/names_devi
        NL_ind:         index into NameLookup file.
```

Note that the type and NL_type for a given element may indicate different element types, for example: The vertical trim magnet "utv1" has

```
        type = ATTR_SBEND, and
        NLtype = IS_V_STEER.
```

This is because in the lattice definition, "utv1" has a non-zero design value in order to aim the survey coordinates correctly from the AGS to RHIC.

- examine

Table: *bl_football*

```
The table "bl_football" is just a template for an SDS object for the o
from the "bl" program.  The column names are pretty self-descriptive
of the 21 unique elements of the beam hyperellipsoid matrix.  The 15 e
of the lower triangle are not included, since the matrix is symmetric.
```

- examine

Table: *bl_lattices*

```
The table "bl_lattices" contains information for constructing a lattic
model for the "bl" program from various lattice and Namespace SDS file
(usually from the Holy_Lattice area).  A "model" is a particular group
of beamline elements to be appended in a prescribed order.  A given mo
may have multiple entry lines, given with a monotonically increasing
"sequence" number.  The columns are used as follows:

        model:          model name for a particular part of the
                        lattices of various machines or beam lines
                        to be concatenated together,
        sequence:       a monotonic entry number for this model,
        holy:           flag for location of source of this segment
                        of the model (=0 for current path, =1 for
                    Holy_Lattice, =2 for a tcelemnt from the
                    tcelement table.),
        lattice:    file name of the lattice SDS file or a tceleme
                    name,
        Namespace:  file name of the Namespace file or blank for
                    a tcelement,
        first:      SiteWideName of first element to use or blank
                    if from the beginning or end of the file,
        last:       SiteWideName of last element to use or blank
                    if from the beginning or end of the file,
        direction:  =1 for forward ordering of this segment or
                    =-1 to reverse the order.
```

- examine

Table: *bl_model_names*

```
The table "bl_model_names" is just a template for an SDS object for th
output from the "bl" program.  The column names are:

        model_name:     name of the model used from the "bl_lattices"
        scheme_name:    name of the scheme.
        db_name:        name of the database containing the "blinit" g
```

- examine

Table: *bl_monitors*

```
The table "bl_monitors" lists beamline elements where output from the
"bl" program should be produced.  There can be several schemes of moni
and tweakable elements for a given model.  The columns are as follows:

        model:          model name of a model defined in "bl_lattices"
        scheme:         a key to a group of tweakable elements within
                        model,
        monitor:        SiteWideName of an element used as a monitor.
```

- examine

Table: *bl_orbit*

```
The table "bl_orbit" is just a template for an SDS object for the
output from the "bl" program.  The column names are:

        x:      x of trajectory calculated at element,
        xp:     x' of trajectory calculated at element,
        y:      y of trajectory calculated at element,
        yp:     y' of trajectory calculated at element,
        z:      z of trajectory calculated at element,
        u:      dp/p of trajectory calculated at element.
```

- examine

Table: *bl_tweaks*

```
The table "bl_tweaks" lists beamline elements that can be tweaked with
a model.  There can be several schemes of tweakable elements and monit
for a given model.  The columns are as follows:

        model:          model name of a model defined in "bl_lattices"
        scheme:         a key to a group of tweakable elements within
                        model,
        element:        a SiteWideName of an element which can be twea
```

- examine

Table: *bl_twiss*

```
The table "bl_twiss" is just a template for an SDS object for the outp
from the "bl" program.  The column names are pretty self-descriptive
of Twiss and dispersion functions, as well as phase advances.
```

- examine

Table: *blm_avg*

The table "blm_avg" lists the average sensitivity and slope for each BLM signal line. These are averaged over all the cans connected to the signal line. The columns are as follows:

```
        signal:         SiteWideName of the BLM signal.
        sensitivity:    sensitivity [pA/R/Hr] at 1400V averaged over a
                         attached to this signal line.
        sigma:          standard deviation of the sensitivity averaged
                         all cans attached to this signal line.  Here
                         sigma = sqrt[ (<x**2>-<x>**2) * n/(n-1) ].
        slope:          slope of sensitivity with voltage [pA/R/hr/kV]
```

```
                    at 1400V averaged over all cans attached to t
                    line.
```

- examine

Table: *blm_HV*

The table "blm_HV" keeps track of how the high voltage is wired for the BLM cans. The columns are as follows:

```
HV:     High voltage line.
can:    SiteWideName of a can connected to this high voltage l
```

- examine

Table: *blm_serial*

The table "blmser" links the SerialName from the "atr_cal" database table "blm_calib" to a given site wide name location in the tunnel. The columns are as follows:

```
can:            SiteWideName of the BLM can location in the tu
                These column entries match "can" entries in th
                tables "blm_HV" and "blm_wire", as well as the
                "SiteWideName" column of the table "othereleme
SerialName:     Serial name of the BLM can from the "atr_cal"
                database.
```

- examine

Table: *blm_wire*

The table "blm_wire" links the BLM signal lines to individual ionization chambers (cans). The columns are as follows:

```
signal: SiteWideName of the signal line.
can:    SiteWideName of the ionization chamber.
```

- examine

Table: *BodyHarm*

The table "BodyHarm" contains magnet measurment data from the Magnet Division for magnets in the ATR. The date is from the short coils for fields in the body of the magnet (no end effects). This table is identical in from to the "BodyHarm" table for the cryogenic RHIC magnets. The columns are as follows (some descriptions are incomplete):

Magnet:	Id of magnet.
ColdMass:	
RunNum:	Data run number for measurements.
TestDate:	Date of measurments.
MeasCoil:	probe Id.
Element:	
RefRadius:	reference radius for multipoles.
Analysis:	
Currnt:	current in amperes.
UpDown:	direction of ramp.
WarmCold:	warm for all ATR magnets.
a0-&>;a10:	skew multipoles.
b0-&>;b10:	normal nultipoles.
TransFunc:	transfer function B/I? for dipoles and G/I? fo
FieldAngle:	
FldAngVar:	
FldAngSTD:	
Notes:	
LoginName:	
ModDateTime:	

- examine

Table: *bpm_orientation*

The table "bpm_orientation" has information about how a particular bpm has been installed. The columns are:

SerialName:	serial name of the bpm
top:	port in the top position
left:	port in the left position (beam's eye)
bottom:	port in the bottom position
right:	port in the right position (beam's eye)

- examine

Table: *Connectable*

Connectable is not listed in the "table_desc" table.

- examine

Table: *Connects*

Connects is not listed in the "table_desc" table.

- examine

Table: *ConnectType*

ConnectType is not listed in the "table_desc" table.

- examine

Table: *Contains*

Contains is not listed in the "table_desc" table.

- examine

Table: *Device*

Device is not listed in the "table_desc" table.

- examine

Table: *DeviceType*

DeviceType is not listed in the "table_desc" table.

- examine

Table: *fid_offsets*

```
The "fid_offsets" table contains a list of fiducials and their offsets
relative to the element center for a given element or type of elements
```

```
The columns are as follows:

   ftname:      name of group of fiducial offsets which links back to
                SerialName in the "magnet_data" table,
   fidname:     name of fiducial to be combined with a SurveyName to
                generate a specific fiducial survey name as used by th
                Survey and Alignment Group,
   dx:          horizontal transverse offset of the fiducial,
   dy:          vertical offset,
   dz:          longitudinal offset.
```

- examine

Table: *generic_bpm_info*

```
The table "generic_bpm_info" contains some generic information about
bpm types.  Its columns are:

        GenericName:    the generic name of a bpm,
        nplanes:        how many active planes in the design,
        id:             inner diameter in meters.
```

- examine

Table: *generic_size*

```
The table "generic_size" provides a set of outline dimensions for disp
elements along the beamline.  Its columns are:

        GenericName:            GenericName of element
        xoff:                   transverse horizontal offset of center
        yoff:                   vertical offset of center
        zoff:                   longitudinal offset of center
        dx:                     width of element
        dy:                     height of element
        dz:                     length of element
        pen:                    pen number for drawing pictures

Here the element is assumed to be a rectangular box.
```

- examine

Table: *header_groups*

```
The "header_groups" and "header_tables" tables define groups of tables
```

```
for use with the dbapplications tools.  The entries in the "header_gro
table are:

        groupname:      the name of a group of related tables,
        filename:       filespecs for writting  a C header file
                        with the utility "db2gh",
        comment:        a short description of the group (255 chars
                        or less).

Type "man dbgroups" from owl.rhic.bnl.gov for more information.
```

- examine

Table: *header_tables*

```
The "header_groups" and "header_tables" tables define groups of tables
for use with the dbapplications tools.  The entries in the "header_tab
table are:

        groupname:      the name of a group of related tables,
        sequence:       a number for ordering the tables within a grou
                        This must be unique for tables within a partic
                        group.  The order should be such that tables c
                        be written without causing trigger problems.
        tablename:      the name of a table within the group.
        dbname:         the name of the database in which to find the
                        table.  At present this should be just the (de
                        current database.
        host:           This is for future expansion with different se
        prefix:         a prefix to be added to the SDS structure name
                        when there may be a conflict with other tables
                        from other databases.

Type "man dbgroups" from owl.rhic.bnl.gov for more information.
```

- examine

Table: *htmldoclines*

```
The table "htmldoclines" contains some lines to be included in the HTM
documentation output by the "describe_doc_html" stored procedure.  The
columns are as follows:

        ind:            an index name for a part of the file.  At pres
                        only "head", "begin", "end" will be used for
                        the  section, beginning of  section,
                        and end of the  section, respectively.
        sequence:       a sequence order for lines within an index sec
        line:           the text to be output.
```

- examine

Table: *installation*

```
The "installation" table contains information about installation of co
in the tunnel.  The structure of this table changes from time to time
work is done.  Currently the columns are as follows (I hope.):

        name:           SiteWideName of component,
        fids_defined:   Fiducials have been defined for the surveyors.
        In_Tunnel:      The component is located on its stand within
                        the tunnel.
        Surveyed:       The component has been surveyed, if necessary.
        Bussed:         Major power bussing for magnets has been compl
                        in the tunnel.
        Vacuum:         The beampipe has been pumped down for this com
        Water:          The water connections have been made for this
                        component.
        ctrl_wires:     Other control/sensor wires have been connected
                        the component.
        SA_test:        Have stand alone tests been completed.

This table is updated every few days, so some things may not be quite
date.
```

- examine

Table: *Integral*

The table "Integral" contains magnet measurment data from the Magnet Division for magnets in the ATR. The date is from the long probe coils for fields including the ends of the magnets. This table is identical in from to the "Integral" table for the cryogenic RHIC magnets. The columns are as follows (some descriptions are incomplete):

```
        Magnet:         Id of magnet.
        ColdMass:
        BNLorVend:
        RunNum:         Data run number for measurements.
        TestDate:       Date of measurments.
        MeasCoil:       probe Id.
        Element:
        RefRadius:      reference radius for multipoles.
        Analysis:
        Currnt:         current in amperes.
        UpDown:         direction of ramp.
        WarmCold:       warm for all ATR magnets.
        a0-&>;a10:      skew multipoles.
        b0-&>;b10:      normal nultipoles.
        TransFunc:      transfer function Bl/I for dipoles and Gl/I fo
        FieldAngle:
        Notes:
        LoginName:
        ModDateTime:
```

- examine

Table: *madc_channel*

madc_channel is not listed in the "table_desc" table.

- examine

Table: *magnet_data*

```
The "magnet_data" table contains information specific to an actual ser
number of an element.  These entries are joined to the "magnet_slot" t
through the SerialName column.  The columns are as follows:

   SerialName:          actual serial name of a particular device,
   ftname:              link into "fid_offsets" table for generating
                        fiducials relative to the IP coordinates,
   FieldName:           link into a the "magnet_field" table for trans
                        function information of a particular kind of m
   long_corr:           a longitudinal correction which allows for a s
                        misalignment of the fiducial template used in
                        construction of some dipole magnets,
   top_halfcore:        serial name of the top halfcore of a dipole ma
   bottom_halfcore:     serial name of the bottom halfcore of a dipole
```

- examine

Table: *magnet_field*

The table "magnet_field" contains transfer function information for the the various types of magnets in the ATR lines. The columns are as follows:

```
FieldName:      key to select a given type of magnet data
sequence:       sequence number for data
I:              current in amperes
Transfunc:      Bl/I for dipoles, Gl/I for quads
RefRadius:      reference radius for multipole components (in
UpDown:         direction of ramp
                        =0 for simulation,
                        =+1 for up,
                        =-1 for down
b0:             Normal dipole multipole (=1 for dipoles, =0 fo
b1:             Normal quadrupole multipole (=1 for quads)
b2:             Normal sextupole component
b3:             Normal octopole component
b4:             Normal decupole component
b5:             Normal 12-pole component
a0:             Skew dipole component
a1:             Skew quadrupole multipole (=1 for quads)
a2:             Skew sextupole component
a3:             Skew octopole component
a4:             Skew decupole component
a5:             Skew 12-pole component
```

- examine

Table: *magnet_field_bak*

magnet_field_bak is not listed in the "table_desc" table.

- examine

Table: *magnet_slot*

```
The "magnet_slot" table contains information for individual beamline l
of elements, such as magnets, flags, bpm's, etc.  It does not contain
tion on vacuum components.  The colums are as follows:

   SiteWideName:        SiteWideName of the element,
   GenericName:         GenericName of the element, such as a model na
   SerialName:          Actual serial id of what element is installed,
   Orientation:         This specifies the direction of installation o
                        element: +1 for forward, -1 for reversed.  The
                        elements currently reversed are some dipoles.
   sag_correction:      A transverse offset of magnet's center from de
                        trajectory to account for bending angles.
   IP_fixed:            A flag for indicating that the IP has been def
```

- examine

Table: *movableBLM*

The table "movableBLM" contains a list of BLM can names which are considered movable and can be moved via the worldwideweb form. The columns are as follows:

```
        SiteWideName:   SiteWideName of the BLM can.
```

- examine

Table: *NameLookup*

The "NameLookup" table lists the elements ATR lattice designs and associates disparate but equivalent names for each of these elements. This is a combination of the Holy_Lattice Namespace files:

```
        $HOLY_LATTICE/YTransfer/Namespace  and
        $HOLY_LATTICE/BTransfer/Namespace
```

The C-header file $HORST/include/gddb/db_NameLookup.h describes the SDS version of this table as dumped to

```
       $HOLY_LATTICE/ATR_common/db_NameLookup.sds
```

The function db_NameLookup_in_sds() reads builds an SDS object from the SDS file. The columns are:

```
        lattice_index:  a pointer into the lattice object of the SDS f
        atom_index:     a pointer into the Twiss and Survey files
        fid_index:      not used for ATR data
        network_index:  not used at present
        type:           an integer device type code as defined by the
                        $HORST/include/names_devicetypes.h
        orientation:    +/-1 indicating direction of installation in t
        Machine:        ="ATR" for the ATR.
        InOut:          not used for ATR.
        Section:        ='U', 'W;, 'X', or 'Y' for the corresponding b
        DeviceName:     mnemonic for type of device (see below).
        DevNo:          a sequence number for devices within a beamlin
        SiteWideName:   the SiteWideName (see below).
        SurveyName:     a corresponding name for the surveyors.
        SerialName:     serial name of installed component.
        LatticeName:    name used in the lattice description.
        GenericName:    a model name for the device.
        CoordinateType: specifier for the type of coordinates
        Scoord:         "s"-coordinate along the beam line of center o
                        device, generally in the center of the element
        Sequiv:         same as Scoord for the ATR.
        Ncoord:         RHIC N (north) coordinate.
        Wcoord:         RHIC W (height) coordinate.
        Ecoord:         RHIC E (east) coordinate.
        theta:          azimuthal direction of beam relative to the E-
        phi:            vertical pitch angle of beam.
        psi:            roll angle about the beam.
```

- mnemonic list from "common_gddb..DevMnemonic" table.
- SiteWideName description.

- examine

Table: *nominal_bends*

```
The table "nominal_bends" contains the nominal bend angle for bend mag
such as dipoles, pitches, lambertsons, and the switch magnet.  Its col

        SiteWideName:   Site wide name of the  element
        InOut:          This is blank for the ATR
        Section:        Beamline name in upper case ('U', 'W', 'X' or
        Machine:        ='ATR'
        angle:          the bend angle in radians

Note that InOut and Machine are in this table for compatability with t
RHIC rings.  (SiteWideName, InOut, Section, Machine) form a primary ke
the NameLookup table if everything gets lumped into a single table in
future.
```

- examine

Table: *otherelement*

The table "otherelement" contains information about tunnel elements which are not listed in the "NameLookup" table. At present, this includes vacuum components, BLM's, and the beam dump. The columns are as follows:

```
        SiteWideName:   SiteWideName of the element.
        GenericName:    generic model name of the element.
        ds:             downstream distance (s) from middle of beam li
                        element "dsof" (negative values are upstream).
        dsof:           SiteWideName of an reference element for deter
```

```
                  of 3d coordinates.
dx:               horizontal transverse displacement of element
                  design trajectory.
dy:               vertical displacement of element from design t
```

- examine

Table: *ps_data*

The table "ps_data" contains a description of actual power supplies.
The columns are as follows:

psserial:	serial name for the actual power supply in thi
model:	model name of the type of power supply,
V_rating:	maximum operating voltage,
I_rating:	maximum current,
polarity:	polarity information,
Int_card:	type of integration card (digital/analog),
regulation:	fractional regulation level.

- examine

Table: *ps_estimate*

The values in the "ps_estimate" table are estimates of what the actual
operating currents might be. They were used in calculating power requ
for the ATR. The columns are as follows:

SiteWideName:	site wide name of the power supply,
Vdc:	estimated operating voltage,
Idc:	estimated operating current.

Clearly, the trim magnet values should be much less on average.

- examine

Table: *ps_installation*

The table "ps_installation" tracks installation of magnet power suppli
for the ATR. Most entries are 0 for incomplete, or 1 for completed.
At present the columns are:

name:	SiteWideName of the power supply,
finished:	construction complete,

```
        in_house:       moved into house,
        bussed:         power bus connections made,
        water:          water connections made, if any,
        dac:            dac installed.
```

- examine

Table: *ps_limits*

```
The table "ps_limits" contains other operational limits for the magnet
power supplies.  The rows are as follows:

        SiteWideName:   site wide name of the power supply,
        Max_I_safety:   a current limit for safety considerations (suc
                        for keeping the coils from overheating, or for
                        radiation considerations),
        Min_I_safety:   a minimum limit for safety considerations.  Th
                        current should stay between the "Max_I_safety"
                        and "Min_I_safety" values for safe operation.
        Max_I_advice:   an advisory limit for maximum current,
        Min_I_advice:   an advisory limit for minimum current,
```

- examine

Table: *PS_Mag_Wireup*

PS_Mag_Wireup is not listed in the "table_desc" table.

- examine

Table: *ps_ref*

ps_ref is not listed in the "table_desc" table.

- examine

Table: *ps_slot*

```
The table "ps_slot" contain a list of magnet power supplies for the AT

        psserial:       serial name for the actual power supply in thi
        SiteWideName:   site wide name for a power supply slot,
        house:          location of the power supply slot.
```

- examine

Table: *quad_transf*

```
The table "quad_transf" is under development
```

- examine

Table: *RealContains*

RealContains is not listed in the "table_desc" table.

- examine

Table: *RealDevices*

RealDevices is not listed in the "table_desc" table.

- examine

Table: *Spigot*

Spigot is not listed in the "table_desc" table.

- examine

Table: *SpigotOwn*

SpigotOwn is not listed in the "table_desc" table.

- examine

Table: *SWNameAdo*

```
The table "SWNameAdo" lists ADO’s which are associated with a particul
```

```
SiteWideName.  The columns are:

        SiteWideName:   Site wide name of a particular device/element,
        ADOName:        Name of a particular ADO instance.
```

- examine

Table: *SWNameAdo_bak*

SWNameAdo_bak is not listed in the "table_desc" table.

- examine

Table: *SWNdesc*

The table "SWNdesc" has quick descriptions (up to 80 chars) of various things which have SiteWideName's. The columns are as follows:

```
        SiteWideName:   SiteWideName of the thing.
        type:           type of SiteWideName keys into "SWNtable"
        comment:        The definition or description.
```

- examine

Table: *SWNtable*

The table "SWNtable" connects a SiteWIdeName listed in the table "SWNdesc" through "SWNdesc.type" to the appropriate table where more information may be found. The columns are as follows:

```
        type:           type of SiteWideName keys into "SWNtable"
        tablename:      name of table to look for information.
```

- examine

Table: *table_desc*

```
This table gives comment lines for describing the various tables in th
database.  It has the following three columns:
   name:  the name of a table in the database,
   num:   an ordered line sequence number for the comment
   line:  the actual comment line (up to 80 characters).
The combination of (name, num) form a unique clustered index.
```

- examine

Table: *tcelement*

```
The table "tcelement" contains information about trajectorily challeng
elements.  This table may eventually be moved into the optics database
"atr" for the ATR.  It is in a prototype stage to replace some old opt
tables which point to SDS files in the Holy_Lattice directory.  The ta
contains coefficients of Taylor series up to 6th order.  The columns a

        tcename:        a name of a tcelement,
        ind0:           0th order index,
        ind1:           1st order index,
        ind2:           2nd order index,
        ind3:           3rd order index,
        ind4:           4th order index,
        ind5:           5th order index,
        ind6:           6th order index,
        value:          value of coefficient.

Only nonzero coefficients need to be specified. Currently indices run
0 to 5 for nonzero values.  An entry of 255 (tinyint) should be used f
indices higher than needed for a particular coefficient.
```

- examine

Table: *tunnel_arcs*

The table "tunnel_arcs" contains some drawing information for the "plot_atr" program. Each row contains the 2-D information for drawing an arc in the "D" projection. The columns are as follows:

```
        x0:     The "E" coordinate of the center of curvature.
        y0:     The "N" coordinate of the center of curvature.
        r:      The radius of the arc
        theta0: The angle of the beginning endpoint.
        theta1: The angle of the final endpoint.
```

Note that theta0 and theta1 are measured in degrees counterclockwise from the parallel to the E-axis.

- examine

Table: *tunnel_arcs_bak*

tunnel_arcs_bak is not listed in the "table_desc" table.

- examine

Table: *tunnel_lines*

The table "tunnel_lines" contains some drawing information for the "plot_atr" program. Each row contains the 2-D endpoints of a line segment to be plotted in the "D" projection. The columns are as follows:

```
        x0:     The "E" coordinate of the first endpoint.
        y0:     The "N" coordinate of the first endpoint.
        x1:     The "E" coordinate of the second endpoint.
        y1:     The "N" coordinate of the second endpoint.
```

- examine

Table: *tunnel_lines_bak*

tunnel_lines_bak is not listed in the "table_desc" table.

- examine

View: *bl_model_info*

```
The view "bl_model_info" is just a template for an SDS object for the
output from the "bl" program.  The output just mirrors the information
of the model from the "bl_lattices" table. The column names are:

        holy:           flag for location of source of this segment
                        of the model (=0 for current path, =1 for
                        Holy_Lattice, =2 for a tcelemnt from the
                        tcelement table.),
        lattice:        file name of the lattice SDS file or a tceleme
                        name,
        Namespace:      file name of the Namespace file or blank for
                        a tcelement,
        first:          SiteWideName of first element to use or blank
                        if from the beginning or end of the file,
        last:           SiteWideName of last element to use or blank
                        if from the beginning or end of the file,
        direction:      =1 for forward ordering of this segment or
                        =-1 to reverse the order.
```

- examine

View: *halfcores*

The view "halfcores" provides a list of serial names for the halfcores
in ATR dipoles. Its columns are:

```
        SiteWideName:           SiteWideName of dipole,
        SerialName:             SerialName of installed dipole,
        top_halfcore:           serial name of top halfcore,
        bottome_halfcore:       serial name of bottom halfcore.
```

- examine

View: *MAD_type*

The view "MAD_type" is an example of a view from tables in two differe
databases: "NameLookup" from the "atr_gddb" database and "magnet_piece
from the "atr" lattice optics definition database. The columns are:

```
        SiteWideName:   SiteWideName of the element from "NameLookup"
        type:           the name of an element type used by the "MAD"
                         program
```

The SQL create statement for this view was:

```
  create view MAD_type as
     select SiteWideName, atr..magnet_piece.type
       from NameLookup, atr..magnet_piece
       where LatticeName=atr..magnet_piece.name
```

- examine

View: *magfield*

The view "magfield" is a view of relevant pieces of the "magnet_field" table. The columns are as follows:

```
        FieldName:      key to select a given type of magnet data
        sequence:       sequence number for data
        UpDown:         direction of ramp
                                =0 for simulation,
                                =+1 for up,
                                =-1 for down
        RefRadius:      reference radius for multipole components (in
        I:              current in amperes
        Transfunc:      Bl/I for dipoles, Gl/I for quads
        b0:             Normal dipole multipole (=1 for dipoles, =0 fo
        b1:             Normal quadrupole multipole (=1 for quads)
```

- examine

View: ***magnet_survey***

```
The view "magnet_survey" is a view of selected columns from the tables
"magnet_slot" and "magnet_data".  This view is used by the program
"atrfid" to generate survey fiducials for the beamline elements.  Its
columns are as follows:

   SiteWideName:        SiteWideName of the element,
   ftname:              link into "fid_offsets" table,
   Orientation:         flag for reversed dipole stands,
   long_corr:           a longitudinal correction which allows for a s
                        misalignment of the fiducial template used in
                        construction of some dipole magnets,
   IP_fixed:            A flag for indicating that the IP has been def
```

- examine

View: ***magquick***

The view "magquick" links the SiteWideName to lattice_index and FieldName for magnets in the ATR. The columns are as follows:

```
        SiteWideName:   SiteWideName of the magnet from NameLookup.
        lattice_index:  Pointer into the Holy_Lattice flat SDS file.
        FieldName:      Key into the "magquick" view for the transfer
                        function of the magnet.
        Scoord:         The s-coordinate from the beginning of the U-l
```

- examine

View: *names*

```
The view "names" gives a subset of columns from the "NameLookup" table
with easier names to type.  The columns are:

        n:              SiteWideName of element,
        s:              the Scoord of the element,
        lat:            LatticeName of the element,
        gen:            GenericName of the element,
        survey:         SurveyName of the element,
        dev:            DeviceName of the element,
        devno:          DevNo of the element.
```

- examine

View: *pi_NameL*

```
The view "pi_NameL" is a narrower view of "NameLookup" with a little b
the "magnet_slot" table.  The columns are:
        SiteWideName:   SiteWideName of element.
        Section:        section name (beam line) of element.
        GenericName:    generic name of the element.
        orientation:    the orientation: forward (+1) or reversed (-1)
        CoordinateType: the type of coordinates (IP, Mech ctr, Traj ct
        Scoord:         the position along the beamline.
        Ncoord:         the N (~north) coord.
        Wcoord:         the W (height) coord.
        Ecoord:         the E (~east) coord.
        theta:          the azimuthal angle.
        phi:            the pitch angle.
        psi             the roll angle.
        sag_corr:       a shift of the element center from the given c
                        (This is primarily for dipoles.)
```

- examine

View: *ps_view*

```
The view "ps_view" is a useful combination of information from the
"ps_basic" group of tables.  The rows are as follows:

        SiteWideName:   site wide name of the power supply,
        psserial:       serial name for the actual power supply in thi
        polarity:       polarity information,
        V_rating:       maximum operating voltage,
        I_rating:       maximum current,
        Max_I_safety:   a current limit for safety considerations (suc
                        for keeping keeping the coils from overheating
                        for radiation considerations.),
        Min_I_safety:   a minimum limit for safety considerations.  Th
                        current should stay between the "Max_I_safety"
                        and "Min_I_safety" values for safe operation.
        Max_I_advice:   an advisory limit for maximum current,
        Min_I_advice:   an advisory limit for minimum current,
        Int_card:       type of integration card (digital/analog),
        regulation:     fractional regulation level.
```

- examine

View: *ufoil*

The view "ufoil" contains the s-coordinate location of the stripping foil relative to the beginning of the U-line. It has one entry:

```
Scoord:         the s-coordinate from the beginning of the U-l
```

- examine

Procedure: *dbg2sds_all*

The procedure "dbg2sds_all" generates a separate line with a dbg2sds command for each group in the atr_gddb database. The output from this procedure may be piped into a shell to generate SDS files for all groups in this database.

Procedure: *describe*

```
The stored procedure "describe" reads a description from the "table_de
table for a database object if the description exists.  Its use is as

        describe $name

where $name is a database object from the sysobjects table.
```

Procedure: *fill_blm_avg*

The procedure "fill_blm_avg" truncates the table "blm_avg" and regenerates the averages of sensitivties and slopes for the BLM cans on each signal line. The callibration data is from the "atr_cal..blm_calib" table, and table joins are made to the signal via data stored in the "atr_gddb..blm_wire" and "atr_gddb..blm_serial" tables.

Procedure: *html_describe_doc*

```
The procedure "html_describe_doc" generates documentation in an HTML f
```

Procedure: *html_get_doclines*

```
The procedure "html_get_doclines" selects lines of HTML code from the
"htmldoclines" table for a given "ind" value.  It takes as a single
argument, the "ind" value as search key.
```

Procedure: *html_group_list*

```
The procedure "html_group_list" lists (in an HTML format) the groups o
tables and views defined by the "header_info" group.
```

Procedure: *html_gt_list*

```
The procedure "html_gt_list" generates a series of lists (in HTML form
of tables and views in all the groups.
```

Procedure: *html_list_anchor*

```
The stored procedure "html_list_anchor" generates a listed HTML anchor
reference.  Its syntax is
        htmlanchor
where  is the name of the anchor to reference, and
 is the text to be displayed.
```

Procedure: *html_p_list*

```
The procedure "html_p_list" generates a description in HTML format for
eadh procedure using data in the "table_desc" table.
```

Procedure: *html_procedure_list*

```
The procedure "html_procedure_list" generates a list (in HTML format)
of all procedures.
```

Procedure: *html_t_list*

```
The procedure "html_t_list" generates a description in HTML format for
eadh table and view using data in the "table_desc" table.
```

Procedure: *html_table_describe*

```
The stored procedure "html_table_describe" reads a description from th
table for a database object if the description exists.  Its use is as

        html_table_describe $name

where $name is a database object from the sysobjects table.
```

Procedure: *html_ungrouped_list*

```
The procedure "html_ungrouped_list" generates a list (in HTML format)
of all tables and views not in a group.
```

Procedure: *initserial*

```
The procedure "initserial" updates the "magnet_slot" and "magnet_data"
tables by changing the SerialName entries in both tables to a new valu
will only do this if the initial value of the SerialName looks like a
SiteWideName (ie., if it begins with a lower case 'u', 'w', 'x', or 'y
The syntax is:

  initserial $SWN, $newSN

where $SWN is the SiteWideName of the element, and $newSN is the new S
```

Procedure: *installed*

```
The stored procedure "installed" gets installation information about t
from the "installation" and "magnet_slot" tables in the atr_gddb datab
Its use is as follows:

        installed $pat

where $pat is an SQL string matching pattern for a SiteWideName.

The returned columns indicate the following:

        name:           SiteWideName of the element
```

```
IP:             = 1 if an IP has been defined for this element
                = 0 if not
fids:           Have fiducials been defined for this element?
Tunnel:         Has the element been installed in the tunnel?
Surveyed:       Has the element been surveyed?
Bussed:         Has the main power bus been connected to this
                magnet.
ctl_wires:      Have the other wires been connected to this de
Vacuum:         Has the vacuum system been pumped down and lea
                checked for this device?
Water:          Have the cooling water connections been comple
SA_test:        Has the stand alone test been completed for th
                device?
```

- execute

Procedure: *set_ps_limits*

The procedure "set_ps_limits" is a temporary procedure which was used to set up some quasi-almost-questionably reasonable (maybe/maybe not) values in the "ps_limits" table. This will most likely be deleted in the future.

Procedure: *undocumented*

```
The procedure "undocumented" lists tables, views, and procedures which
which have not been entered into the "table_desc" table.
```

Procedure: *what*

```
The stored procedure "what" finds and gives some information about obj
(tables, views, rules, etc.) and/or columns by matching a patern to en
in the system tables for the atr_gddb database.  The system stored pro
"sp_help" and "sp_helptext" can be quite useful for finding out about
objects listed in the "sysobjects" table.  The usage for "what" is:

         what $pat

where $pat is an SQL pattern for matching a string.
```

Procedure: *whatsit*

```
The procedure "whatsit" gives the column names, and types for table/vi
which match a pattern.  The usage is
```

```
        whatsit $pat
where $pat is a table/view name or an SQL pattern.
```

About this document:

The actual HTML code is written by the SYBASE server when this page is invoked, so things should be reasonably up to date, with the exception of some of the structure definitions of tables under development. Each of the tables and views has an **examine** link, which may be used to view the data currently stored in the database table(s). (An actual database querry is sent when the **examine** link is invoked.

Tables and views are generally grouped into **group**s of related tables. A view is a kind of window which looks at selected data from various tables within the database. A view does not have any extra data, but may present data from various tables and other views in a different way. (Data from other databases may even be included in a view.) (return status = 0)

5.2 Calibration database, atr_cal

$/www.rhichome.bnl.gov/cgi - bin/atr\_cal\_describe.sh$

Document generation time: Oct 11 1995 4:14PM

Description of the "atr_cal" database.

This document shows what is currently in the database. (Some of the text descriptions of things may be old, if the the appropriate tables have not been updated.)

Quick index:

- Groups of tables and views
- Ungrouped tables and views
- Stored procedures

Groups of tables and views:

```
There are several useful tools for dealing with groups of tables.
(See the local UNIX man pages for "dbtools".)

Note to programmers:
Header files (for C and C++) can be found along the "$HORST/include" p
for most of these groups.  (The "header_info.h" file is located along
"$DBAPPLICATIONS/include" path.)  By including these paths in your
compilations, you should get the file (if it has been generated by "db
Functions for reading and writing SDS versions of the groups are locat
in "$HORST/lib/$ARCH/libgddb.a".
```

- header_info
- html
- table_desc

Ungrouped tables and views:

- blm_calib
- BodyHarm
- Integral
- magnames

Stored procedures:

- html_describe_doc
- html_get_doclines
- html_group_list
- html_gt_list
- html_list_anchor
- html_p_list
- html_procedure_list
- html_t_list
- html_table_describe
- html_ungrouped_list
- undocumented
- what
- whatsit

Group *header_info*

The C header file is "Dbapps/header_info.h".

This lists how tables are collected into associated groups in *.h files.

The tables and views in this group are:

- header_groups
- header_tables

Group *html*

The C header file is "gddb/html.h".

This group contains some information for generating HTML documentation from the database.

The tables and views in this group are:

- htmldoclines

Group *table_desc*

The C header file is "gddb/table_desc.h".

A table for describing database tables.

The tables and views in this group are:

- table_desc

Table: *blm_calib*

The table "blm_calib" contains calibration information for the BLM ionization chambers. The columns are as follows:

```
        SerialName:     The serial name of the can.
        calib-date:     The date of calibration (ignore time).
        sensitivity:    Measured sensitivity in [pA/R/hr] at a positiv
                        bias of 1400V.
        slope:          Measured slope in [pA/R/hr/V] at a positive bi
                        of 1400V.
        comment:        Up to 80 characters of comments.
```

- examine

Table: *BodyHarm*

The table "BodyHarm" contains magnet measurment data from the Magnet Division for magnets in the ATR. The date is from the short coils for fields in the body of the magnet (no end effects). This table is identical in from to the "BodyHarm" table for the cryogenic RHIC magnets. The columns are as follows (some descriptions are incomplete):

```
        Magnet:         Id of magnet.
        ColdMass:
        RunNum:         Data run number for measurements.
        TestDate:       Date of measurments.
        MeasCoil:       probe Id.
        Element:
        RefRadius:      reference radius for multipoles.
        Analysis:
        Currnt:         current in amperes.
        UpDown:         direction of ramp.
        WarmCold:       warm for all ATR magnets.
        a0-&>;a10:      skew multipoles.
        b0-&>;b10:      normal nultipoles.
        TransFunc:      transfer function B/I? for dipoles and G/I? fo
        FieldAngle:
        FldAngVar:
        FldAngSTD:
        Notes:
        LoginName:
        ModDateTime:
```

- examine

Table: *header_groups*

```
The "header_groups" and "header_tables" tables define groups of tables
for use with the dbapplications tools.  The entries in the "header_gro
table are:

        groupname:      the name of a group of related tables,
        filename:       filespecs for writting  a C header file
                        with the utility "db2gh",
        comment:        a short description of the group (255 chars
                        or less).

Type "man dbgroups" from owl.rhic.bnl.gov for more information.
```

- examine

Table: *header_tables*

```
The "header_groups" and "header_tables" tables define groups of tables
for use with the dbapplications tools.  The entries in the "header_tab
table are:

        groupname:      the name of a group of related tables,
        sequence:       a number for ordering the tables within a grou
                        This must be unique for tables within a partic
                        group.  The order should be such that tables c
                        be written without causing trigger problems.
        tablename:      the name of a table within the group.
        dbname:         the name of the database in which to find the
                        table.  At present this should be just the (de
                        current database.
        host:           This is for future expansion with different se
        prefix:         a prefix to be added to the SDS structure name
                        when there may be a conflict with other tables
                        from other databases.

Type "man dbgroups" from owl.rhic.bnl.gov for more information.
```

- examine

Table: *htmldoclines*

```
The table "htmldoclines" contains some lines to be included in the HTM
documentation output by the "describe_doc_html" stored procedure.  The
columns are as follows:

        ind:            an index name for a part of the file.  At pres
                        only "head", "begin", "end" will be used for
                        the  section, beginning of  section,
                        and end of the  section, respectively.
        sequence:       a sequence order for lines within an index sec
        line:           the text to be output.
```

- examine

Table: *Integral*

The table "Integral" contains magnet measurment data from the Magnet Division for magnets in the ATR. The date is from the long probe coils for fields including the ends of the magnets. This table is identical in from to the "Integral" table for the cryogenic RHIC magnets. The columns are as follows (some descriptions are incomplete):

```
        Magnet:         Id of magnet.
        ColdMass:
        BNLorVend:
        RunNum:         Data run number for measurements.
        TestDate:       Date of measurments.
        MeasCoil:       probe Id.
        Element:
        RefRadius:      reference radius for multipoles.
        Analysis:
        Currnt:         current in amperes.
        UpDown:         direction of ramp.
        WarmCold:       warm for all ATR magnets.
        a0-&>;a10:      skew multipoles.
        b0-&>;b10:      normal nultipoles.
        TransFunc:      transfer function Bl/I for dipoles and Gl/I fo
        FieldAngle:
        Notes:
```

```
        LoginName:
        ModDateTime:
```

- examine

Table: *magnames*

The table "magnames" gives a correspondence between the magnet measurements and actual serial names of magnets. The columns are as follows:

```
        SerialName:     Serial name of magnet listed in atr_gddb..magn
        Magnet:         The Magnet Division's name of the measured mag
```

Most of the dipoles were measured without vacuum chambers. After measurments these magnets were taken apart and reassembled with vacuum chambers. Not all magnets were measured.

- examine

Table: *table_desc*

```
The table "table_desc"gives comment lines for describing the various
tables in the database.  It has the following three columns:

        name:  the name of a table in the database,
        num:   an ordered line sequence number for the comment
        line:  the actual comment line (up to 80 characters).

The combination of (name, num) form a unique clustered index.
```

- examine

Procedure: *html_describe_doc*

```
The procedure "html_describe_doc" generates documentation in an HTML f
```

Procedure: *html_get_doclines*

```
The procedure "html_get_doclines" selects lines of HTML code from the
"htmldoclines" table for a given "ind" value.  It takes as a single
argument, the "ind" value as search key.
```

Procedure: *html_group_list*

```
The procedure "html_group_list" lists (in an HTML format) the groups o
tables and views defined by the "header_info" group.
```

Procedure: *html_gt_list*

```
The procedure "html_gt_list" generates a series of lists (in HTML form
of tables and views in all the groups.
```

Procedure: *html_list_anchor*

```
The stored procedure "html_list_anchor" generates a listed HTML anchor
reference.  Its syntax is
        htmlanchor
where  is the name of the anchor to reference, and
 is the text to be displayed.
```

Procedure: *html_p_list*

```
The procedure "html_p_list" generates a description in HTML format for
eadh procedure using data in the "table_desc" table.
```

Procedure: *html_procedure_list*

```
The procedure "html_procedure_list" generates a list (in HTML format)
of all procedures.
```

Procedure: *html_t_list*

```
The procedure "html_t_list" generates a description in HTML format for
eadh table and view using data in the "table_desc" table.
```

Procedure: *html_table_describe*

```
The stored procedure "html_table_describe" reads a description from th
table for a database object if the description exists.  Its use is as

        html_table_describe $name

where $name is a database object from the sysobjects table.
```

Procedure: *html_ungrouped_list*

```
The procedure "html_ungrouped_list" generates a list (in HTML format)
of all tables and views not in a group.
```

Procedure: *undocumented*

```
The procedure "undocumented" lists tables, views, and procedures which
which have not been entered into the "table_desc" table.
```

Procedure: *what*

```
The stored procedure "what" finds and gives some information about obj
(tables, views, rules, etc.) and/or columns by matching a patern to en
in the system tables for the atr_gddb database.  The system stored pro
"sp_help" and "sp_helptext" can be quite useful for finding out about
objects listed in the "sysobjects" table.  The usage for "what" is:

         what $pat

where $pat is an SQL pattern for matching a string.
```

Procedure: *whatsit*

```
The procedure "whatsit" gives the column names, and types for table/vi
which match a pattern.  The usage is

        whatsit $pat

where $pat is a table/view name or an SQL pattern.
```

About this document:

The actual HTML code is written by the SYBASE server when this page is invoked, so things should be reasonably up to date, with the exception of some of the structure definitions of tables under development. Each of the tables and views has an **examine** link, which may be used to view the data currently stored in the database table(s). (An actual database querry is sent when the **examine** link is invoked.

Tables and views are generally grouped into **group**s of related tables. A view is a kind of window which looks at selected data from various tables within the database. A view does not have any extra data, but may present data from various tables and other views in a different way. (Data from other databases may even be included in a view.) (return status = 0)

Chapter 6

Safety

6.1 Safety features

/RHIC/ATR/safety/features.html

Last update: 26 June, 1995

Features of the ATR/RHIC safety system

Design and Implementation

- Redundant microswitches on access doors.
- Dual critical devices.
  - Reachback devices disabled if either critical device fails.
- Fail-safe design: fail into a safe state.
- PLC's monitoring system and controlling modes.
  - Two peer groups will be used this fall.
    - Peer 3 for zones: US, U, Wup, VT, VTP, V1, D6, and M.
    - Peer 5 for zones: Wdn, X, and Y.
    - Some elements are monitored by both Peers
      - Specifically WGS1 has four microswitches.
  - Two independently coded programs run in each peer group: either one may force a safer mode.
- The system will be tested with procedures to be approved by the RSC.

Restricted and Controlled Accesses

Only properly trained people will be allowed access to the ATR.

- Controlled access to a specific region:
  - Only one gate will be used for entrance and exit.
  - Simultaneous key and electrical strike release is required.
  - An operator will be stationed by the gate to allow only qualified people to enter.
- Restricted access to a specific region:
  - Special "0" keys similar but not identical to the "256" keys of the AGS will be used.
  - The rules for using these "0" keys similar to the 256 keys.

For detailed information about the system, contact Bob Frankel.

Mangled by Waldo MacKay (waldo@bnl.gov)

6.2 Radiation gates and zones

/RHIC/ATR/safety/gates.html

Radiation gate and zone locations.

Gate names

The names are made up of a beamline name, a two letter gate code, and a sequence number. The gate codes are defined as follows:

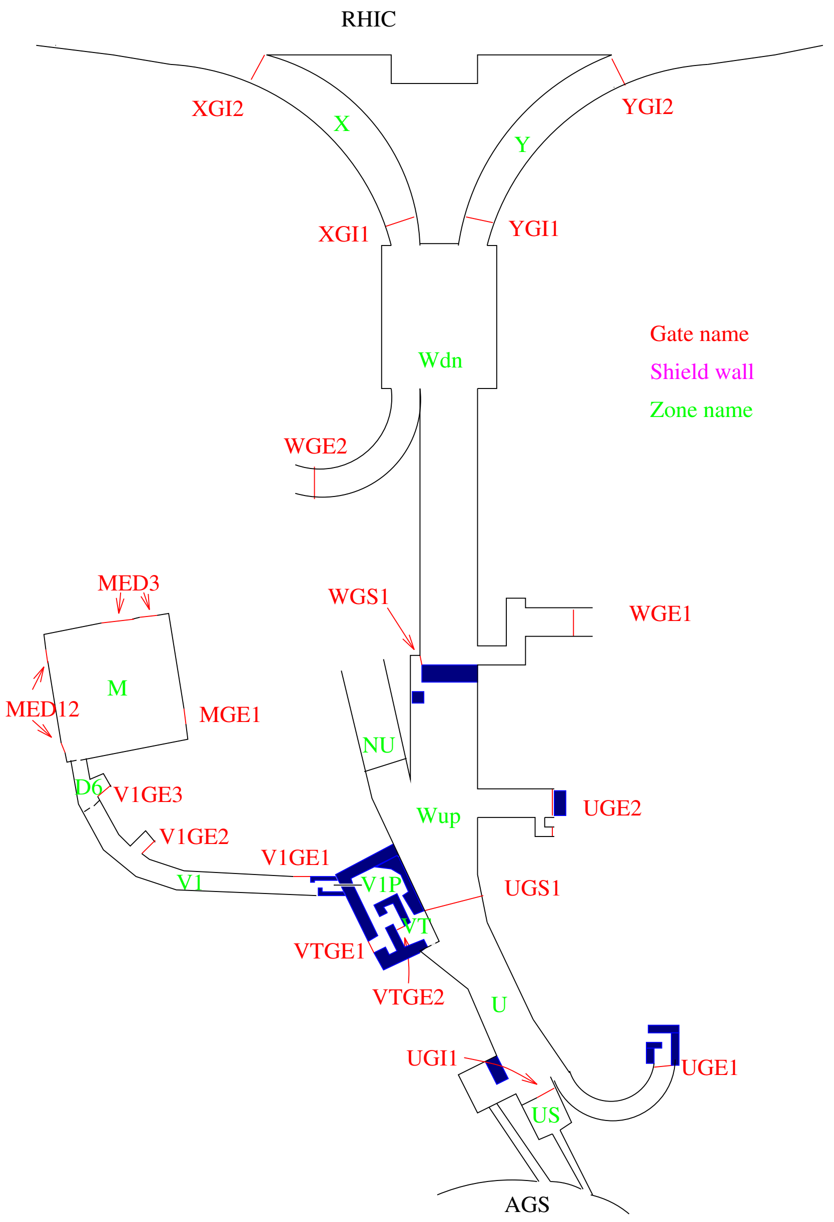
```
ED  Walkout emergency door
GS  Sectionalizing gate
GE  Entry gate
GI  Interlocking gate
```

Zone names

```
US  U-line stub tunnel
U   U-line downstream of stub tunnel
Wup W-line upstream of shield wall
Wdn W-line downstream of shield wall
X   X-line tunnel
Y   Y-line tunnel
VT  Inner g-2 target cave
V1P V1-line primary cave inside V-target blockhouse
V1  V1-line downstream of blockhouse
D6  V1D6 pit at end of V1 line
M   Muon storage ring
```

The gates UGI1 and UGS1 will not be necessary starting next fall, so they will not be interlocked. The zones Wup, U, and US will effectively be a single area as far as sweeps and access are concerned.

Mangled by Waldo MacKay (waldo@bnl.gov)
Last update: 24 June, 1995


RHIC
XGI2
X
Y
YGI2
XGI1
YGI1
Gate name
Shield wall
Zone name
Wdn
WGE2
MED3
WGS1
WGE1
M
MED12
MGE1
NU
D6
V1GE3
Wup
UGE2
V1GE2
V1GE1
V1
V1P
UGS1
VT
VTGE1
VTGE2
U
UGI1
UGE1
US
AGS

6.3 Radiation rate classifications

/RHIC/ATR/safety/radclasses.html

Radiation Security System Classification

General Guideline for AGS Radiation Security System Classification and Application

Allowable Radiation				**Access**		
Radiation Area Name	**AGS Class**	**Whole body absorbed dose rate or dose equivalent rate**	**30GeV Large Beam Fluence Rate [p/cm^2/hr]**	**Access when area is reset**	**Sweep/Reset Authority**	**Enclosures and Gates**
Very High Radiation	I	>500 rad/hr	>3.9x10^9	Absolute Prohibition	Operator or RSC designate	Impregnable enclosures, Interlocks
High Radiation	II	>50 rem/hr <500 rad/hr	>1.1x10^8 <3.9x10^9	Special Procedure	Health Physics or RSC designate	Full enclosure, Interlocks
	III	>5 rem/hr <50 rem/hr	>1.1x10^7 <1.1x10^8	Health Physics Supervision	Health Physics or RSC designate	Walls/fixed fences, Interlocks
	IV	>0.1 rem/hr <5 rem/hr	>2.3x10^5 <1.1x10^7	Authorized Individuals	Authorized Individuals	Walls/fences, Locked gates
Radiation	V	>0.005 rem/hr <0.1 rem/hr	>1.1x10^4 <2.3x10^5	Rad worker or escorted visitor	Not required	Signs every 20', Ropes at perimiters
Control	VI	>0.00005 rem/hr <0.005 rem/hr	>1.1x10^2 <1.1x10^4	GERT trained or escorted	Not required	Signs at entrances

Note: The information in this table comes from *AGS Operation Procedures Manual*: Section 9.1.11 Revision 02 (18 May, 1995).

Mangled by Waldo MacKay (waldo@bnl.gov)
Last update: 26 June, 1995

6.4 Chipmunks

/RHIC/ATR/safety/chipmunks.html

List of Chipmunk Locations along the U, V, W, X, and Y-lines

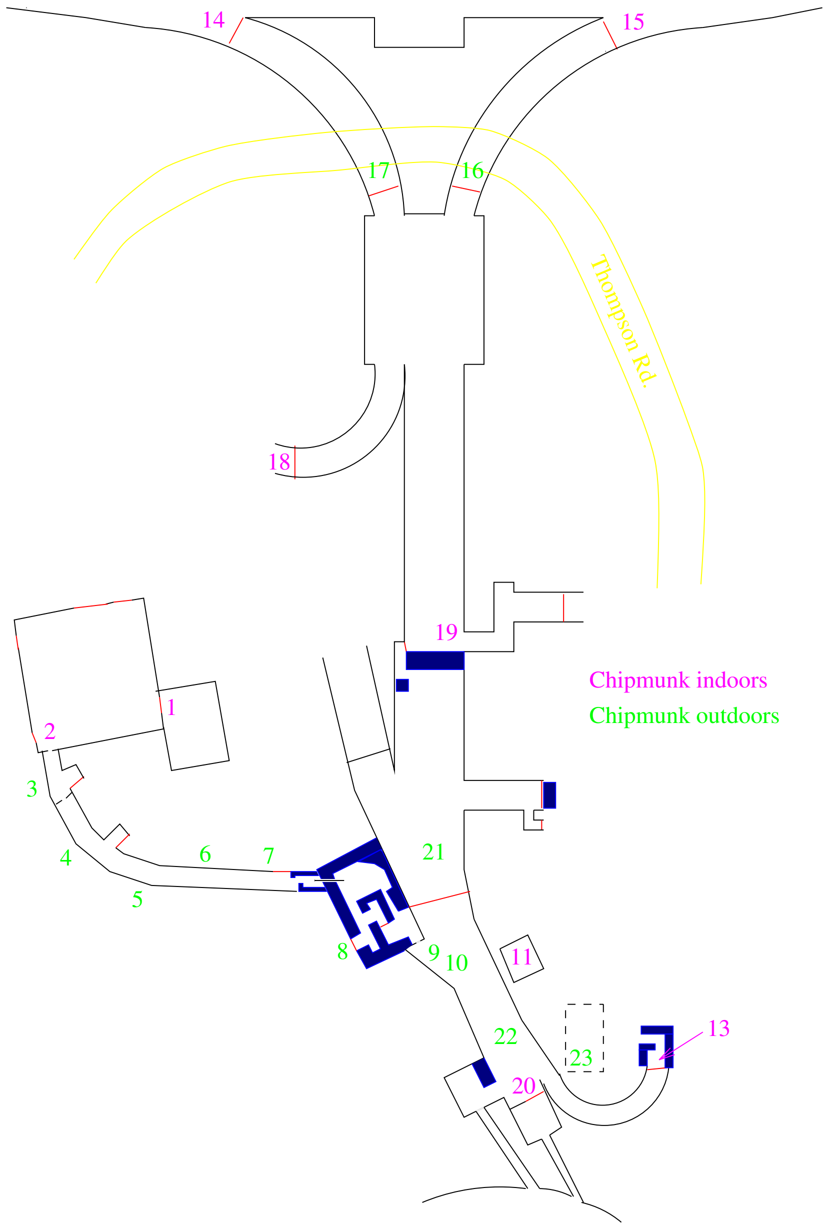
Name	Reset Group	Location
C1	M	inside Muon Ring Control Room
C2	M	inside SW corner of Building 919 (g-2 experiment?)
C3	V1	along V1 tunnel.
C4	V1	along V1 tunnel.
C5	V1	along V1 tunnel.
C6	V1	along V1 tunnel.
C7	V1	along V1 tunnel.
C8	VT	outside gate VTGE1 (outer entrance g-2 target blockhou
C9	VT	on berm downstream of VQ9 (two required).
C10	VT	on berm downstream of VQ9 (two required).
C11	VT	dehumidifier room (igloo).
C13	U	outside gate UGE1 (upstream entrance to U-line).
C14	XY	at gate XGI2 (downstream end of X-line).
C15	XY	at gate YGI2 (downstream end of Y-line).
C16	XY	S edge of Thompson Rd. above Y-line.
C17	XY	S edge of Thompson Rd. above X-line.
C18	W	inside weather door at gate WGE2 (entrance near beam d
C19	W	downstream of W-line shield wall.
C20	U	downstream of U-line stub tunnel.
C21	U	on berm downstream of collimators uc2 and uc3.
C22	U	on berm downstream of collimator uc1.
C23	U	at corner of power substation nearest upstream U-line.
C12	-	spare channel not implemented.

The precise locations of C3 through C7 are yet to be determined.

%

%A postscript version of this figure is available.

Mangled by Waldo MacKay (waldo@bnl.gov)
Last update: 22 June, 1995


14
15
17
16
Thompson Rd.
18
19
Chipmunk indoors
Chipmunk outdoors
1
2
3
4
5
6
7
21
8
9
10
11
22
23
13
20

6.5 Critical devices

/RHIC/ATR/safety/criticaldevs.html

ATR Beamline Access and Critical Devices.

For FEB (fast extracted beam) opertation of the ATR to the beam dump (**wbd**) at the end of the W-line, the beamline zones will be grouped into logical regions for **access** and sweeps.

Fall 1995 running with Gold Ions

Access to ATR upstream of shield wall (Zones: US, U, and Wup)

Film badges are currently required! Currently this region is a radiation *controlled access* region requiring a film badge and activation checks for removing material. During and after next fall's running, the area will become a class I radiation area with beam, and a class IV area with **no beam in the AGS ring**. Critical devices for this area are:

```
Radiation class (beam on):    I (no access)
Radiation class (beam off):   IV
Device #1:                    BF6        (Booster)
Device #2:                    BDH1, BDH2 (Booster)
```

Access to ATR north of shield wall (Zones: Wdn, X, and Y)

These areas are currently unconstrolled areas (class VI) and no film badge is required. Next fall these areas will become controlled areas **requiring badges** .

We would like to have access to the areas downstream of the shield wall (between wd6 and wd7) when beam is being transported to the *g*-2 experiment. In order to guarantee that no radiation is produced downstream of the wall, two devices must be interlocked so that their power supplies are disabled for either *restricted access* or *controlled access*.

When we are running beam to the dump at the end of the W-line, the complete ATR transfer lines must be secured with *access prohibited*. The areas north (downstream) of the shield wall will be able to be put into controlled or restricted access mode by disabling two "critical devices": the *8 degree* and *20 degree* bends.

```
Radiation class (beam on):    I   (no access)
Radiation class (beam off):   IV
Device #1:                    ud3-6 (8 degree bend)
      power source:           psuarc8
      location of supply:     A house
Device #2:                    wd1-8 (20 degree bend)
      power source:           pswarc20
      location of supply:     1000P
```

Access to the *g*-2 inner target cave (Zone: VT)

Access to the *g*-2 inner target cave will be **prohibited** while beam is being transported in the U-line.

```
  Radiation Class (U-line beam) I  (no access)
  Radiation Class (g-2 beam)    I  (no access)
  Radiation Class (no AGS beam) IV? (before Jan '96 g-2 run)
  Radiation Class (no AGS beam) ? (after g-2 run)
                                      to be determined.

  Device #1:                    BF6
  Device #2:                    BDH1, BDH2
```

Access to the *g*-2 outer target cave (V1-primary) (Zone: V1P)

Access to the *g*-2 outer target cave will be **prohibited** while beam extracted into the U-line. Access with no FEB extraction, but with beam in the AGS might be possible, if the RSC agrees.

```
  Radiation Class (U-line beam) I  (no access)
  Radiation Class (g-2 beam)    I  (no access)
  Radiation Class (no FEB beam) IV? (before Jan 96 g-2 run)

  Device #1, and #2:            UD1, UD2  (4 degree bend)
        power source:           psuarc4
                                   two independent contactors
```

Access to *g*-2 beam lines beyond the target blockhouse (Zones: V1, D6, and M)

These areas should be accessable during the Fall '95 FEB tests with heavy ions to the W-line beam dump.

Future considerations for *g*-2, the Sextant test and beyond

These will be determined by the Radiation Safety Committee in the future.

---

g-2 Experiment with High Intensity Protons

Critical Devices for the V1, D6, and M zones will be:

```
  Device #1:                    V1D1 -- V1 dipole
  Device #2:                    V1D2 -- V1 dipole
```

Sextant test

For the sextant test in the fall of 1996, the 4 and 5 o'clock sextant will become a class I radiation area with beam. For access into the RHIC tunnel (4-5 and 6 o'clock sections), the critical devices in the ATR will probably be the switching magnet *swm* and the 90 degree dipole busses in the X- and Y-lines.

```
Device #1:                    swm
      power source:           psswm
      location of supply:     1000P
Device #2 (for X-line side):  xd1-xd31 and xlamb
      power source:           psxarc90
      location of supply:     1000P
Device #2 (for Y-line side):  yd1-yd31 and ylamb
      power source:           psyarc90
      location of supply:     1000P
```

For the sextant test the reversing switch in the switching magnet power supply will need to be disabled so that only one polarity may be used.

Stored beams in RHIC

With stored beams in the RHIC rings, the X and Y zones will need to have *prohibited access* to prevent exposure. Most likely access could be made to zones (Wdn, ...) upstream provided there is no hazard radiation from upstream operations.

Mangled by Waldo MacKay (waldo@bnl.gov)
Last update: 25 June, 1995

6.6 Fault studies

/RHIC/ATR/safety/faultstbl.html

Fault Studies for Fall 1995 Heavy Ion Run to W-dump

#	Loss location	Control Elements	Measurement Location
1	H10 septum	H10 bump	With septum off, what do we see at stub tunnel?
2	ud1,ud2	psuarc4 off	End of zero degree port.
			End of stub tunnel.
3	uq4	psuarc4	Outside gate UGE1.
4	uc1	Jaws: uc1l, uc1r [1]	Outside gate UGE1.
			Corner and floor of substation nearest U-line.
			Along 0-degree line toward g-2 and vertical shaft.
			With chipmunks at WGS1.
5a	ud3 ---> ud5	psuarc8	In igloo.
			Gate VTGE1 (g-2 blockhouse).
			Trenches east and west of line.
5b	uc2	Jaws: uc1l, uc1r [1]	In igloo.
			Gate VTGE1 (g-2 blockhouse).
			Trenches east and west of line.
6	Old Neutrino line.	pswarc20 off.	Entrances and pipes from old neutrino area (Gate #5).
			With chipmunks at WGS1.
7	wp1	pswarc20	Overhead survey shaft and sleeve to east.
8	wd3	pswp1	UGE2, vertical shaft, and equipment room.
9	wd6	pswarc20, pswth1 [2]	10" sleevs to east (wd6 near floor).
			Downstream of shield wall.
10a	wd7	pswarc20, pswth1 [2]	Outside WGE1.
			On top of berm. (This area is thinner than downstream.)
10b	wd1 [3]	psutv7 [2]	10" sleeves to west of tunnel through old neutrino line.
11	wq2	pswarc20	Vertical ventilation shaft west of wp2.

12a	wq6 or swm	pswp2, pswtv6 [2]	WGE2.
			Thompson Road.
			Vertical survey shaft on top of berm downstream of swm.
			Sleeves to 1000P (W) power supply house.
			At end of X, Y-arcs. (Gate XGI2, and YGI2.) [4]
12b	Beam dump "wbd"	Standard condition.	WGE2.
			Thompson Road.
			Vertical survey shaft on top of berm downstream of swm.
			Sleeves to 1000P (W) power supply house.
			At end of X, Y-arcs. (Gate XGI2, and YGI2.) [4]
12c	W-line split.	pswp2, pswth5	Ends of X, Y-arcs while beam is scanned [4].

Notes:

1. Each jaw has a length of 6" of tungsten. The pair of jaws may be overlapped.
2. Some tuning may be required from other elements to generate desired losses.
3. This should be less than the wd6 fault into the 10" sleeves, so may be unnecessary. The wd6 fault study should be performed first.
4. Because of the symmetry, only one arc will be necessary. There will be a chipmunk at the end of each arc to provide continuous monitoring.

Mangled by Waldo MacKay (waldo@bnl.gov)
Last update: 22 June, 1995

Appendix A

Operation Procedure Manuals

Appendix B

Alphabetical List of Sitewide Names